JEUX SUR TO 7

Cet ouvrage a été réalisé sous la direction de Benoît de MERLY.

Nous remercions THOMSON pour ses encouragements à la rédaction de ce guide ainsi que pour la mise à disposition des photographies qui illustrent le premier chapitre.

Première édition

Tome 1 ISBN : 2-904457-05-4

EDIMICRO

DÉPARTEMENT ÉDITIONS DE F.D.S. SARL

121-127, avenue d'Italie, 75013 Paris

Sommaire

ANNEXES

CHAPITRE **1**

Techniques de programmation des jeux

Le livre "Jeux sur TO7" a pour but non seulement de vous proposer de nombreux jeux, tous plus passionnants les uns que les autres, mais aussi de vous faire découvrir les techniques les plus employées pour la programmation de jeux sur TO7. Ce premier chapitre vous montre en particulier comment choisir un bon jeu, comment programmer ce jeu en utilisant une méthode rapide et efficace. Vous trouverez aussi dans ces quelques pages de nombreux conseils et "trucs" qui vous permettront de programmer vous-même de très beaux jeux, amusants à jouer, rapides et bien protégés.

1.1 CHOISIR UN « BON » JEU

Lorsque vous décidez de programmer un jeu sur votre micro-ordinateur, deux solutions s'offrent à vous : choisir un jeu très connu ou bien inventer vous-même un nouveau jeu. Le choix de la première solution n'appelle pas de remarque particulière, nous nous intéresserons plus particulièrement à la seconde solution : création d'un jeu nouveau et "bon".

Il est fort difficile, voire même impossible de définir exactement ce qu'est un bon jeu. Certaines personnes peuvent en effet apprécier un jeu alors que d'autres personnes trouvent ce jeu sans intérêt. Il n'y a donc pas de recette miracle pour fabriquer le jeu idéal qui plaît à tous.

L'expérience montre que l'on peut, tout de même, dresser une liste de caractéristiques qu'un jeu se doit de posséder pour être le plus attrayant possible:

- le jeu doit mettre à l'épreuve le joueur :

  . mise à l'épreuve de son intelligence (jeux de réflexion),

  . mise à l'épreuve de son habilité et de ses réflexes (jeux d'action rapides),

  . mise à l'épreuve de sa chance (jeu de hasard).

- le jeu doit être suffisamment complexe pour retenir le plus longtemps possible l'attention du joueur mais ne doit pas être trop difficile afin de ne pas le décourager rapidement. On se désintéresse en effet très vite d'un jeu trop simple auquel on gagne à chaque fois que l'on joue, de la même façon que l'on se désintéresse d'un jeu auquel on ne parvient jamais à gagner !

- le jeu doit être adapté aux conditions de jeu sur micro-ordinateur. On joue souvent seul ou à deux sur un micro-ordinateur mais il est difficile de jouer à plus. Programmer un jeu de société qui se joue à cinq ou six joueurs, c'est programmer un jeu auquel vous ne jouerez jamais.

- le jeu doit être fait pour le joueur et non pour le micro-ordinateur. Le programmeur qui fait un Mastermind où le joueur choisit une combinaison secrète, que l'ordinateur décode, fait preuve de grandes qualités en matière de programmation mais son beau programme n'est pas un jeu amusant, ce n'est même pas du tout un jeu !

1.2 MÉTHODE DE PROGRAMMATION D'UN JEU

1.2.1 Qualités d'un bon programme

Un programme, qu'il soit de jeu ou non, est bon si-et seulement si-il est correct (aucune erreur ne peut être tolérée), efficace (le cahier des charges doit être scrupuleusement respecté), facilement modifiable (programmation structurée ...) et bien protégé (idiotproof comme disent les anglo-saxons).

Un bon programme de jeu se doit, de plus, de faire bon usage des possibilités graphiques (couleurs, haute résolution, définition de nouveaux caractères) et sonores du micro-ordinateur pour lequel il est écrit. La durée d'exécution est souvent un facteur critique, il faut donc optimiser la programmation : astuces, algorithmes évolués...

1.2.2 Etapes de la programmation

Ecrire un programme possédant toutes les qualités énoncées ci-dessus exige l'utilisation d'une méthode de programmation rationnelle et efficace.

Si vous essayez en effet de taper directement le programme à partir du clavier au fur et à mesure que

vous l'élaborez dans votre tête, vous pouvez être sûr d'être obligé de passer ensuite un temps incroyablement long pour mettre au point ce programme à grands renforts de GOTO, de tests compliqués et inutiles... Vous ne pourrez jamais être certain d'avoir éliminé toutes les erreurs. Il y a en outre peu de chance pour que votre programme fasse exactement ce que vous désirez lui voir faire et toute modification est quasiment impossible à apporter ! Cette démarche absurde est en tous points semblable à celle d'un maçon qui, ayant décidé de construire une maison, commence tout de suite à empiler les briques les unes sur les autres. Ce maçon est ensuite obligé de rajouter des portes et des fenêtres oubliées... Personne ne pourra rentrer en toute confiance dans cette maison sans avoir peur de recevoir quelque chose sur la tête.

De même qu'un bon maçon prend soin de dresser un plan de sa maison et d'en prévoir les moindres détails avant de commencer à la construire, un bon programmeur doit d'abord écrire son programme sur le papier en suivant les quatre étapes fondamentales suivantes :

- analyse
- dessin d'un organigramme
- codage
- mise au point

L'analyse et le dessin d'un organigramme sont à un programme ce qu'est le plan de l'architecte pour une maison. Le codage et la mise au point correspondent à la construction de la maison et aux finitions.

L'ANALYSE

L'analyse est l'une des plus importantes étapes de la programmation. C'est elle en effet qui permet de dresser le cahier des charges complet que le programme doit respecter.

Une fois le jeu choisi ou inventé suivant les critères donnés précédemment, il vous faut définir précisément le but du jeu, ses règles, sa présentation, son début, sa fin, le rôle du ou des joueurs...

DESSIN D'UN ORGANIGRAMME

Cette étape de la programmation correspond au dessin de la structure du programme en fonction des résultats obtenus par l'analyse. Au cours de cette étape, plusieurs blocs distincts seront définis afin d'obtenir la programmation la plus structurée possible. Cela permet ensuite d'écrire et de mettre au point chaque bloc séparément. La structure modulaire ainsi obtenue permet de plus de faire ultérieurement des modifications ou des améliorations.

On peut dans cette étape remettre en cause le cahier des charges donné par l'analyse : est-on capable de faire tel ou tel bloc, une contrainte due à la durée d'exécution prévisible d'un bloc ne permet peut-être pas de réaliser telle ou telle fonction (tir de plusieurs missiles en même temps, déplacements simultanés de plusieurs objets...).

Exemple d'application :

- Programme CHENILLE

  . programme principal

- sous-programme de présentation
- sous-programme d'entrée des noms des joueurs
- sous-programme de jeu
- sous-programmes utilitaires

LE CODAGE

Les étapes précédentes étant maintenant terminées, il s'agit de transcrire les résultats obtenus dans un langage compréhensible pour le micro-ordinateur. Chaque bloc est donc écrit en BASIC en prenant soin de numéroter les lignes de 10 en 10 afin de pouvoir éventuellement insérer ensuite de nouvelles lignes.

Il est fortement recommandé de bien structurer au niveau de la numérotation des lignes un programme, cela donne une meilleure lisibité au programme et facilite bien les choses pour retrouver une erreur.

Exemple :

- 0-1000 bloc 1
- 1000-2000 bloc 2

\- Programme CHENILLE

0-1000 programme principal

1000-2000 sous-programme de présentation

2000-3000 sous-programme d'entrée des noms des joueurs

3000-4000 sous-programme de jeu

4000-5000 sous-programme utilitaire

5000-6000 sous-programme utilitaire

LA MISE AU POINT

Cette dernière étape est souvent nécessaire car toutes les précautions du monde n'empêchent pas toujours quelques erreurs. On fait donc tourner le programme bloc par bloc (en insérant des instructions STOP entre chaque bloc par exemple) et on regarde ce qui se passe. Les messages d'erreur ou le comportement bizarre du programme vous permettront de trouver les modifications à faire pour corriger les erreurs.

Si vous adoptez la méthode qui vient de vous être présentée, vous vous rendrez compte très vite que vous programmez très facilement et efficacement de beaux programmes bien structurés et sans erreurs !

1.3 QUELQUES TECHNIQUES PARTICULIÈRES

1.3.1 La présentation

Une jolie présentation est indispensable pour rendre un jeu attrayant. Un jeu qui n'est pas beau à voir est un mauvais jeu. Faire une belle présentation demande du travail, de l'imagination et cela prend beaucoup de place dans le programme (parfois le tiers ou la moitié du programme !) mais le résultat en vaut la peine.

Le TO7 a des possibilités graphiques et sonores qui permettent au programmeur de réaliser tout ce qu'il désire.

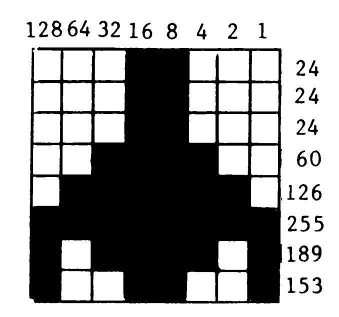
LE CHOIX DES COULEURS

Les instructions SCREEN, COLOR et PSET sont très utilisées pour choisir les couleurs de l'écran, des caractères affichés, ...

DEFINITION DE CARACTERES GRAPHIQUES

L'instruction DEFGR$(I) permet de définir un caractère nouveau sur une matrice 8 x 8.

Exemple :



Ce caractère peut donc être obtenu en faisant :

DEFGR$(0) = 24, 24, 24, 60, 126, 255, 189, 153

Pensez à réserver de la mémoire avec l'instruction CLEAR avant de définir de nouveaux caractères.

POSSIBILITES SONORES

Le TO7 est équipé d'un générateur de sons qui permet de jouer toutes les notes sur une plage de cinq octaves. La durée et l'attaque des notes sont réglables, ainsi que le tempo.

Vous trouverez dans le livre "Guide du TO7" des exemples très détaillés de mise en oeuvre - pour la programmation de jeux - des possibilités du TO7 évoquées ci-dessus.

1.3.2 La rapidité

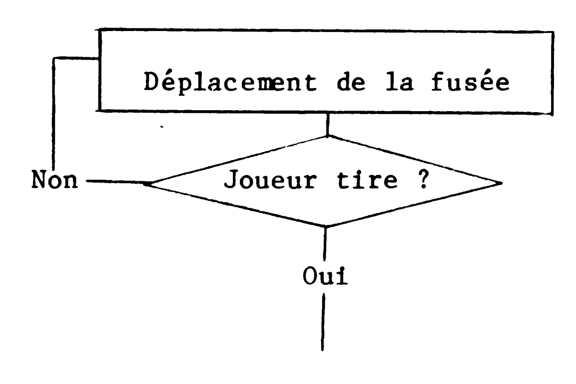
Le temps d'exécution est souvent un facteur critique dans un programme de jeu en BASIC : déplacement trop lent de la balle dans un jeu de mur de briques, déplacement trop lent d'un missile, d'une voiture ...

Il ne faut pas croire que seule la programmation en langage machine est capable de résoudre ce problème. S'il est vrai que la programmation en assembleur élimine quasiment ce problème, il est possible en étant habile de programmer en BASIC des jeux qui tournent vite.

Pour les jeux d'action rapides, le principe est le suivant : le jeu doit être constitué d'une boucle qui doit être la plus courte possible. On ne sort de cette boucle que sur intervention du joueur, par exemple tir d'un missile.

Exemple :

- Programme de jeu MISSILES



Cela permet d'obtenir un déplacement très rapide de la fusée. Chaque fois que le joueur presse la touche "F" pour lancer un missile, la fusée s'arrête et le sous-programme de tir est appelé. Dès que le tir du missile est terminé, il y a retour dans la boucle pour continuer le déplacement de la fusée.

Certains prétendent à tort que le facteur temps n'est pas critique pour les jeux de réflexion. Il est vrai que ce facteur est en général moins important que pour les jeux d'action rapides mais il importe tout de même d'écrire des programmes qui tournent en un temps raisonnablement court : jouer avec un jeu d'Othello où l'ordinateur met dix minutes pour jouer son coup devient vite pénible ! La solution à ce problème est alors l'utilisation d'algorithmes puissants, d'astuces qui permettent d'éviter des tests inutiles.

Pensez à utiliser des variables entières et à mettre plusieurs instructions par ligne. Le gain de temps ainsi obtenu n'est pas négligeable.

Il est bon aussi d'utiliser plusieurs variables plutôt qu'un tableau,

```
XA = 1                             X(1) = 1
XB = 0  est plus rapide que        X(2) = 0
XC = 10                            X(3) = 10
```

Si malgré tous vos efforts votre programme reste lent, il faudra vous résoudre à passer au langage machine ! Les programmes de jeu que vous trouverez dans ce guide vous donneront une bonne idée de ce que l'on peut obtenir en matière de rapidité pour des jeux en BASIC. Des explications détaillées sur la structure de chaque programme sont données. Lisez-les, vous y trouverez peut-être des idées.

1.3.3 La protection

Vous avez certainement constaté que lorsque vous présentez un de vos programmes à vos amis, ceux-ci, consciemment ou non, essayent de faire "planter" ce programme : essayer de faire sortir la raquette du terrain de jeu, entrer une lettre quand l'ordinateur attend un chiffre, donner un nombre négatif et décimal alors que c'est un entier qui est demandé, ...

Il s'agit de faire un programme protégé (idiot-proof).

- Rajoutez des tests qui empêchent la raquette ou la voiture de sortir du terrain de jeu. Cela ralentit bien sûr le jeu mais c'est bien préférable à un message d'erreur.

- Le TO7 refuse toute lettre lorsqu'il attend un nombre mais l'introduction d'un nombre négatif alors que c'est un entier naturel qui est attendu (par exemple : initialisation du générateur aléatoire), peut conduire à des résultats surprenants. Nous vous proposons ci-dessous un petit programme qui permet, lorsque le facteur temps n'est pas critique, d'éviter toute surprise.

```
10  INPUT "ENTREZ UN NOMBRE", NOMB$
20  NOMB = VAL(NOMB$)
30  IF NOMB = 0 THEN 10
40  NOMB = INT (ABS(NOMB))
```

L'intérêt de protéger un programme n'est pas d'empêcher quelqu'un qui le désire vraiment, de "planter" un programme. Cette personne n'a qu'à couper directement l'alimentation du TO7 plutôt que d'essayer de mettre en défaut le programme ! La protection est surtout nécessaire pour empêcher un joueur de faire une faute irréparable en tentant de rattraper une balle ou en essayant d'éviter au dernier moment un astéroïde qui surgit devant lui.

1.3.4 Déplacement d'un mobile

Beaucoup de jeux utilisent les déplacements de différents mobiles : balle, raquette, missile, voiture... Il est donc important de savoir faire se déplacer correctement un mobile sur l'écran.

Pour recréer l'illusion du mouvement, une méthode d'animation couramment utilisée consiste à afficher un caractère correspondant au mobile à un emplacement donné. Après un court laps de temps, un blanc (espace) est mis à cet emplacement pour effacer le caractère. Ce court laps de temps a généralement permis de calculer les nouvelles coordonnées du caractère, soit automatiquement (cas de la fusée dans le jeu MISSILES), soit à partir des données entrées par le joueur sur le clavier ou avec le joystick (cas de la chenille dans le jeu CHENILLE). On affiche alors le caractère à sa nouvelle position et on recommencera ainsi de suite, i mage après image comme dans un dessin animé.

La vitesse de déplacement du mobile dépend évidemment de la distance entre deux positions successives de ce mobile. Cette distance ne doit pas être trop grande pour que le mouvement ne paraisse pas saccadé.

Il faut aussi que le caractère reste visible suffisamment longtemps pour éviter l'impression de clignotement de l'image. Cela dépend de l'intervalle de temps entre deux apparitions du mobile.

Exemple d'application :

```
10   CLS : INPUT "PAS" ; PAS
20   IF PAS = 0 THEN STOP
30   INPUT "DUREE" ; DUREE
40   CLS : LOCATE 0, 0, 0
50   FOR I = 0 TO 39 STEP PAS
60   PSET (I, 10) "*", 0
70   FOR J = 1 TO DUREE : NEXT J
80   PSET (I, 10) " "
90   NEXT I
100  GOTO 10
```

Utilisez ce programme en prenant des valeurs comprises entre 1 et 5 pour PAS et 1 à 200 pour DUREE. Nous vous laissons le soin de trouver vous-mêmes les paramètres qui vous semblent convenir le mieux.

REMARQUE : Le rôle de la ligne 40 est d'effacer l'écran et de supprimer le curseur.

Supposons maintenant qu'un joueur doive déplacer sur l'écran une petite étoile à l'aide des quatre touches : ↑ , ↓ , → et ← .

Le programme pourrait être le suivant :

```
10   CLS : LOCATE 0, 0, 0 : X = 0 : Y = 0
20   PSET (X, Y) "*", 0
30   HX = X : HY = Y
40   A$ = INKEY$ : IF A$ = " " THEN 40
50   A = ASC(A$)
60   IF A = 8 THEN X = X - 1 : GOTO 100
70   IF A = 9 THEN Y = X + 1 : GOTO 100
80   IF A = 10 THEN Y = Y + 1 : GOTO 100
90   IF = 11 THEN Y = Y - 1
100  PSET (HX, HY) " "
110  GOTO 20
```

Rappel :

← a pour code ASCII : 8
→ a pour code ASCII : 9
↑ a pour code ASCII : 10
↓ a pour code ASCII : 11

Il est possible d'améliorer grandement ce petit programme en utilisant une propriété intéressante du TO7 en matière d'évaluation d'expressions logiques.

Le TO7 donne en effet la valeur -1 à une expression logique "vraie" et la valeur 0 à une expression logique fausse.

Essayez, par exemple, :

```
A = 0
PRINT (A = 0)
```

Le résultat est alors -1.

```
PRINT (A = 3)
```

Le résultat est alors 0.

de même :

```
PRINT (A > 3) donne 0
PRINT (A < 3) donne -1
```

Cela permet d'écrire le programme suivant :

```
10   CLS : LOCATE 0, 0, 0 : X = 0 : 4 = 0
20   PSET (X, Y) "*", 0
30   HX = X : HY = Y
40   A$ = INKEY$ : IFA$ = " " THEN 40
50   A = ASC(A$)
```

```
60   X = X + (A = 8) - (A = 9)
70   Y = Y + (A = 11) - (A = 10)
80   PSET (HX, HY) " "
90   GOTO 20
```

Vous remarquerez que ce programme, bien que plus rapide que le programme précédent, possède toujours le défaut de provoquer des messages d'erreur lorsque l'étoile franchit les bords de l'écran. Alors que la protection était difficile à faire dans le programme précédent et que cela aurait agi fâcheusement sur la rapidité du programme, la protection est ici très simple à faire.

Changez les lignes 60 et 70 en :

```
60   X = X + (A = 8 AND X > 0) - (A = 9 AND X
      <39)
70   Y = Y + (A = 11 AND Y > 0) - (A = 10 AND
     Y < 24)
```

Il vous sera désormais impossible de faire sortir l'étoile de l'écran !

Nous vous laissons le soin de modifier vous-même ce programme afin que l'étoile continue à se déplacer dans sa direction précédente si aucune touche n'est appuyée.

1.3.5 Utilisation du joystick

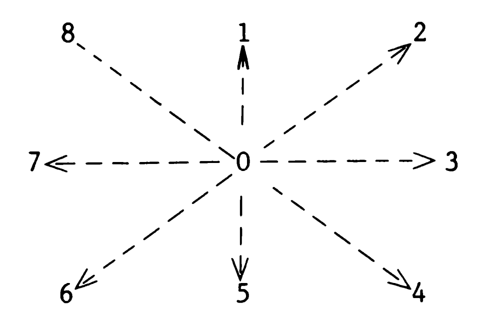
Le TO7 peut être équipé, grâce à un contrôleur spécialement prévu à cet effet, de deux manettes de jeu.

De nombreux jeux d'action utilisent les joysticks car la lecture du port d'entrée - sortie sur lequel ceux-ci sont branchés est beaucoup plus rapide que la saisie du clavier. Cela permet donc de faire des programmes qui "réagissent" plus rapidement aux ordres du joueur.

L'utilisation des joysticks se fait avec deux instructions : STICK(I) et STRIG(I) ou I peut prendre les valeurs 0 et 1.

INSTRUCTION STICK(I)

Suivant la position de la manette de jeu, cette fonction prend les valeurs suivantes :



INSTRUCTION STRIG(I)

Cette fonction donne l'état du bouton de la manette de jeu I.

Elle prend donc :

la valeur 0 si le bouton n'est pas enfoncé,

la valeur -1 si le bouton est pressé.

L'instruction STRIG permet donc de se servir du bouton comme d'une gachette pour déclencher un tir...

Donnons une version de programme de déplacement utilisant le joystick :

```
10   CLS : LOCATE 0, 0 , 0
20   X = 0 : Y = 0 : B = 3
30   PSET (X, Y) "*", 0
40   HX = X : HY = Y
50   A = STICK(1) : IF A = 0 THEN A = B
60   X = X + (A = 7 AND X > 0) - (A = 3 AND X
     <39)
70   Y = Y + (A = 1 AND Y > 0) - (A = 5 AND Y
     <24)
80   PSET (HX, HY) " " : B = A
90   IF STRIG(1) = -1 THEN 500
100  GOTO 30
500  CLS : END
```

Ce programme vous permet de déplacer une petite étoile sur l'écran. Celle-ci garde son ancienne direction si vous lâchez le joystick.

Pour arrêter le jeu, pressez le bouton de la manette de jeu.

1.3.6 Initialisation du générateur aléatoire

La fonction RND(X) donne un nombre réel, pseudo-aléatoire, compris entre 0 et 1, 0 inclu et 1 exclu. De nombreux jeux font appel à cette fonction pour simuler l'intervention du hasard : lancé de dé, choix d'une combinaison secrète, ...

Malheureusement cette fonction donne la même séquence de nombres aléatoires à chaque exécution du programme. Il faut donc remédier à cela en initialisant grâce au joueur le générateur de nombres aléatoires au début du jeu : cela consiste à faire tourner N fois le générateur avant de commencer le jeu, N étant choisi par le joueur.

Le programme correspondant est le suivant :

```
10  LOCATE 0,  : INPUT "ENTREZ UN NOMBRE
    QUELCONQUE", N
20  N = INT (ABS(N))
30  FOR I = 1 TO N
40  Y = RND
50  NEXT I
```

Chaque valeur de N correspond à une séquence de nombres aléatoires. Le joueur peut donc, s'il le désire, rejouer avec la même séquence en donnant toujours la même valeur à N.

REMARQUE : La plupart des présentations de nos jeux utilisent le générateur de nombres aléatoires avant son initialisation. Cela permet d'obtenir toujours la même séquence de nombres qui sont utilisés pour choisir des couleurs, des positions. On évite ainsi d'avoir à initialiser de nombreuses variables pour atteindre le même effet.

1.3.7 Fin de partie

Tous nos jeux se terminent de la même façon afin que les joueurs n'hésitent pas entre telle ou telle touche pour rejouer et arrêter le jeu.

Fin de partie :

```
5000  LOCATE 0, 24, 0 : INPUT "VOULEZ-VOUS
      REJOUER" ; REP$

5010  IF LEFT$(REP$, 1) = "O" OR REP$ = " "
      THEN RUN (ou GOTO n)

5020  IF LEFT$(REP$, 1) = "N" THEN CLS : END

5030  GOTO 5000
```

Seules les touches O et N suivies de n'importe quelle autre touche sont acceptées. Une simple pression de la touche [ENTREE] relance le jeu.

CHAPITRE **2**

Jeux de hasard

2.1 JEU DU 21

La majeure partie du chapitre d'initiation au BASIC par la création d'un jeu, dans le livre "Guide du TO7", est consacrée à la création du jeu du 21. Vous trouverez dans ce chapitre une description détaillée de la façon de procéder pour aboutir au programme final : analyse-organigramme-codage-mise au point.

La version que nous vous proposons ici est une version légèrement modifiée et améliorée au niveau de la présentation par rapport à la version du Guide du TO7.

Rappelons les règles du JEU DU 21.

Au départ vous possédez mille francs. Après avoir misé la somme que vous voulez, vous pouvez lancer le dé autant de fois que vous le désirez pour que votre total de points se rapproche le plus possible de 21, sans toutefois dépasser cette valeur.

Pour lancer le dé, pressez simplement la touche [ENTREE]. Lorsque vous jugerez votre total suffisant, entrez une lettre ou un chiffre quelconque avant de presser [ENTREE]. Le TO7 lance alors le dé pour son propre compte.

Le TO7 détermine ensuite le gagnant et, selon le cas, augmente ou diminue votre avoir... Une nouvelle partie est ensuite lancée.

Si par malheur votre avoir s'annulait, le jeu s'arrête (il vous est en effet impossible de miser par manque d'argent).

Il est interdit de miser une somme plus grande que l'avoir. Une mise nulle arrête la partie en cours.

Après la ruine ou l'arrêt du jeu par introduction d'une mise nulle, une nouvelle partie vous est proposée : pressez soit la touche [O], soit simplement la touche [ENTREE] pour rejouer. La touche [N] arrête le jeu définitivement.

Structure du programme

Programme principal : lignes 10 à 530

Ligne 10, réservation de place mémoire pour deux caractères définis, appel du sous-programme de présentation.

> REMARQUE : L'instruction CLEAR ne peut pas être mise dans un sous-programme car le TO7 perd tous ses pointeurs en exécutant cette instruction. Mis dans l'impossibilité de revenir au programme principal, le TO7 affiche alors un message d'erreur.

Ligne 20, le joueur se voit crédité d'un avoir de mille francs.

Ligne 30, boucle de temporisation.

Ligne 40, effacement de l'écran - initialisations diverses (JOUEUR est le total des points du joueur, MICRO est le total du TO7, K est une variable utilisée pour la présentation des lancés du TO7).

Ligne 50, affiche l'avoir du joueur.

Ligne 60, demande la mise.

Ligne 70, protection pour éviter que l'introduction d'un nombre négatif ou décimal ne provoque une erreur.

Ligne 80, toute mise supérieure à l'avoir du joueur est refusée.

Ligne 90, une mise nulle renvoie à la fin de la partie.

Ligne 100, le joueur veut-il lancer le dé ?

Ligne 110, si le joueur répond à la question précédente par autre chose qu'une simple pression de [ENTREE], le TO7 va alors lancer le dé pour son propre compte.

Ligne 120, lancé du dé pour le joueur - appel du sous-programme de sonorisation.

Ligne 130, affichage du total du joueur.

Ligne 140, si le total du joueur dépasse 21, le joueur a perdu.

Ligne 150, retour à la ligne 100 pour éventuellement un nouveau lancé.

Ligne 160, le TO7 lance le dé pour son propre compte.

Ligne 180, affichage du total du TO7.

Ligne 190, saut de deux lignes en prévision du prochain affichage. Si on est au bas de l'écran, on se replace au début pour éviter un SCROLLING indésirable.

Ligne 200, si le total du TO7 dépasse 21, le joueur gagne.

Ligne 210, si le total du TO7 dépasse le total du joueur, le joueur perd.

Ligne 220, si le joueur et le TO7 ont obtenu tous les deux un total égal à 21, il y a coup nul.

Ligne 230, renvoie au 160 pour un nouveau lancé de dé du TO7.

Ligne 300, affichage du message de gain.

Ligne 310, l'avoir est augmenté de une à cinq fois la mise.

Ligne 320, retour en 30 pour une nouvelle partie.

Ligne 400, affichage du message de perte.

Ligne 410, diminution de la mise.

Ligne 420, si le joueur possède encore de l'argent, retour en 30 pour une nouvelle partie.

Ligne 430, affichage du message de ruine.

Ligne 500, retour en 30 pour une nouvelle partie.

Sous-programme de sonorisation : lignes 1000 à 1030.

Lignes 1000 à 1020, la gamme chromatique est "montée" très rapidement à dix reprises.

Lignes 1030, retour au programme principal.

Sous-programme de présentation : lignes 2000 à 2120.

Ligne 2000, effacement de l'écran - couleur de l'écran : noir - couleur des caractères : vert

Ligne 2010, caractères en double hauteur et double largeur - affichage du nom du jeu au milieu de l'écran.

Lignes 2020 et 2030, définition de deux caractères représentant des dés à jouer.

Lignes 2040 et 2050, affichage des caractères définis précédemment.

Ligne 2060, initialisation de la variable chaîne P$ avec les notes de la gamme chromatique - l'attaque, le tempo et la durée des notes sont choisis.

Lignes 2070 à 2110, initialisation du générateur de nombres aléatoires.

Ligne 2120, retour au programme principal.

```
10 CLEAR,,2:GOSUB2000
20 AVOIR=1000
30 FORI=1 TO 400:NEXTI
40 CLS:MICRO=0:JOUEUR=0:K=0
50 PRINT"AVOIR: ";AVOIR:PRINT
60 PRINT:INPUT"MISE";MISE
70 MISE=INT(ABS(MISE))
80 IF MISE >AVOIR THEN 30
90 IF MISE=0 THEN 500
100 PRINT:INPUT"JET";J$
110 IF J$<>"" THEN 160
120 JOUEUR=JOUEUR+INT(RND*6+1):GOSUB 1000
130 PRINT"JOUEUR: ";JOUEUR
140 IF JOUEUR>21 THEN 400
150 GOTO 100
160 MICRO=MICRO+INT(RND*6+1)
170 GOSUB 1000
180 LOCATE30,K,0:PRINT"TO7: ";MICRO
190 K=K+2:IF K=24 THEN K=0
200 IF MICRO>21 THEN 300
210 IF MICRO>JOUEUR THEN 400
220 IF MICRO+JOUEUR-42=0 THEN LOCATE15,15,0:PRINT
"COUP NUL !!":GOTO 30
230 GOTO 160
300 LOCATE15,15,0:PRINT"VOUS GAGNEZ!!!"
310 AVOIR=AVOIR+INT(RND*5+1)*MISE
320 GOTO 30
400 LOCATE15,15,0:PRINT"VOUS PERDEZ..."
410 AVOIR=AVOIR-MISE
420 IF AVOIR>0 THEN 30
430 LOCATE15,17,0:PRINT"VOUS ETES RUINE!!!"
500 LOCATE15,22:INPUT"VOULEZ-VOUS REJOUER";REP$
510 IF LEFT$(REP$,1)="O" OR REP$="" THEN 20
520 IF LEFT$(REP$,1)="N" THEN CLS:END
530 GOTO 500
1000 FOR I=1 TO 10
1010 PLAY P$
1020 NEXTI
1030 RETURN
```

```
2000 CLS:SCREEN2,0,0
2010 LOCATE 10,14,0:ATTRB1,1:PRINT"JEU DU 21"
2020 DEFGR$(0)=255,129,165,129,129,165,129,255
2030 DEFGR$(1)=255,129,161,129,129,133,129,255
2040 LOCATE 5,5,0:COLOR 1:PRINT GR$(0)
2050 LOCATE35,20,0:COLOR 5:PRINT GR$(1)
2060 P$="DODO#RERE#MIFAFA#SOSO#LALA#SI":PLAY"A0T
1L404"
2070 ATTRB0,0:LOCATE 0,24,0:COLOR 2:INPUT"ENTREZ
UN NOMBRE QUELCONQUE ",N
2080 N=INT(ABS(N))
2090 FOR I=1 TO N
2100 Y=RND
2110 NEXT I
2120 RETURN
```

2.2 RÉACTION

Avez-vous de bons réflexes ? Le jeu REACTION vous permettra de savoir si vous êtes capable de réagir rapidement.

Une lettre, choisie au hasard parmi les vingt-six lettres de l'alphabet, apparaît soudain en un endroit quelconque de l'écran du téléviseur. Vous devez alors vous précipiter sur le clavier et presser la touche correspondant au caractère affiché !

Le TO7 mesure le temps que vous mettez à réagir et commente à sa façon les résultats obtenus...

Le meilleur temps réalisé depuis que le jeu a commencé est gardé en mémoire par le TO7. Vous pourrez ainsi faire des concours avec vos amis en essayant à chaque fois de pulvériser le précédent record.

Pour rejouer en fin de partie, pressez simplement [ENTREE]. La touche [N] arrête le jeu.

REMARQUE : Le jeu REACTION demande une très bonne connaissance du clavier de la part du joueur. Ne vous faites donc pas trop de soucis au début si vous perdez beaucoup de temps à chercher la bonne touche, l'habitude viendra vite et votre "temps de réaction" diminuera.

Structure du programme

Programme principal : lignes 10 à 200

Ligne 10, appel du sous-programme de présentation.

Ligne 20, mise à zéro du compteur T de temps et choix de la lettre qui va apparaître.

Lignes 30 à 40, boucle de temporisation de durée variable.

Ligne 50, choix de la position où va être placée la lettre.

Ligne 60, affichage de la lettre avec bip sonore.

Ligne 70, le clavier est scruté et le compteur de temps est incrémenté jusqu'à ce que le joueur ait pressé la bonne touche.

Ligne 80, effacement de l'écran.

Ligne 90, affichage du temps de réaction.

Ligne 100, les prochains caractères affichés seront de couleur bleue.

Lignes 110 à 135, messages dont l'apparition ne dépend que du temps de réaction du joueur.

Ligne 140, les prochains caractères affichés seront de couleur violette.

Lignes 150 et 160, suivant le temps mis par le joueur, le record est changé ou non.

Lignes 170 à 200, fin de partie classique.

<u>Sous-programme de présentation</u> : lignes 1000 à 1080

Ligne 1000, effacement de l'écran - couleur de l'écran : noir - couleur des caractères : vert.

Ligne 1001, une boucle de présentation du jeu va être parcourue vingt fois.

Ligne 1002, caractères de taille normale - cette ligne est nécessaire car la taille des caractères est modifiée dans le reste de la boucle.

Ligne 1003, choix d'une position sur l'écran.

Ligne 1004, choix d'une couleur.

Ligne 1005, affichage d'une lettre à la position choisie par la ligne 1003 et de la couleur choisie par la ligne 1004.

Ligne 1006, bip sonore.

Ligne 1008, caractères en double hauteur et double largeur.

Lignes 1009 et 1010, affichage multicolore du nom du jeu.

Ligne 1011, boucle de temporisation.

Lignes 1012 et 1013, on affiche à nouveau le nom du jeu avec de nouvelles couleurs.

Ligne 1014, retour au début de la boucle de présentation (vingt fois).

Lignes 1020 à 1050, initialisation du générateur de nombres aléatoires.

Ligne 1060, effacement de l'écran.

Ligne 1070, initialisation du record avec une valeur très grande.

Ligne 1080, retour au programme principal.

```
10 GOSUB1000
20 T=0:A$=CHR$(INT(RND*26+65))
30 FORI=1TOINT(RND*1000)+100
40 NEXTI
50 X=INT(RND*40):Y=INT(RND*25)
60 LOCATEX,Y,0:PRINT A$:PLAY"L6DO"
70 B$=INKEY$:T=T+1:IF B$<>A$ THEN 70
80 CLS
90 LOCATE4,4,0:PRINT"TEMPS DE REACTION: ";T
100 LOCATE4,7,0:COLOR6,0
110 IF T>100 THEN PRINT"REVEILLEZ VOUS !!!"
120 IF T<30 THEN PRINT"QUELS REFLEXES !!!"
130 IF T<50 AND T>30 THEN PRINT"PAS MAL, VOUS POUV
EZ FAIRE MIEUX..."
135 IF T>200 THEN PRINT "QU'EST CE QUE CA DOIT ETR
E EN VOITURE..."
140 LOCATE7,10,0:COLOR5,0
150 IF T<RECORD THEN PRINT"NOUVEAU RECORD: ";T:REC
ORD=T:GOTO170
160 LOCATE4,10,0:COLOR2,0:PRINT"RECORD TOUJOURS A:
 ";RECORD
170 LOCATE0,20:COLOR 7,0:INPUT"VOULEZ VOUS REJOUER
";REP$
180 IF LEFT$(REP$,1)="O"OR REP$="" THEN CLS:COLOR2
,0:GOTO20
190 IF LEFT$(REP$,1)="N"THEN CLS:COLOR2,0:END
200 GOTO170
1000 CLS:SCREEN2,0.0
1001 FORI=1TO20
```

```
1002 ATTRBO,O
1003 X=INT(RND*40):Y=INT(RND*20)
1004 C=INT(RND*7+1)
1005 LOCATE X,Y,O:COLOR C,O:PRINT CHR$(INT(RND*26+
65))
1006 PLAY"L12DO"
1008 ATTRB1,1:LOCATE10,10,0
1009 COLOR1,O:PRINT"R";:COLOR2,O:PRINT"E";:COLOR3,
O:PRINT"A";:COLOR4,O:PRINT"C";
1010 COLOR5,O:PRINT"T";:COLOR6,O:PRINT"I";:COLOR7,
O:PRINT"O";:COLOR1,O:PRINT"N";
1011 LOCATE10,10,O:FORJ=1TO20:NEXTJ
1012 COLOR7,O:PRINT"R";:COLOR6,O:PRINT"E";:COLOR5,
O:PRINT"A";:COLOR4,O:PRINT"C";
1013 COLOR3,O:PRINT"T";:COLOR2,O:PRINT"I";:COLOR1,
O:PRINT"O";:COLOR7,O:PRINT"N";
1014 NEXTI
1020 LOCATEO,24,O:COLOR2,O:ATTRBO,O:INPUT"ENTREZ U
N NOMBRE QUELCONQUE ",N
1030 FORI=1TON
1040 V=RND
1050 NEXTI
1060 CLS
1070 RECORD=1000
1080 RETURN
```

2.3 TIERCÉ

Inutile d'attendre la fin de chaque semaine pour goûter aux joies des paris et des courses de chevaux. Votre TO7 est en mesure de simuler un tiercé. Tout est réuni pour se trouver dans l'ambiance des courses, seul le commentateur manque !

Le tiercé se joue avec 17 partants ; malheureusement certains ne partent pas lors du grand départ au détriment du joueur qui avait misé ses espoirs sur le cheval récalcitrant.

Chaque joueur dispose au début du jeu d'une somme de 1000 unités. Il est bien sûr impossible de parier au-delà de ses ressources. Les noms des joueurs ont

préalablement été enregistrés et un décompte est affiché à la fin de chaque course. Avant le départ les mises sont enregistrées ainsi que les combinaisons proposées par les joueurs. Les combinaisons invalides sont refusées et le fautif doit réentrer son tiercé.

Toutes les dispositions ayant été prises et chacun pensant gagner, le coup d'envoi est donné par une pression sur la touche [ENTREE].

Après l'arrivée, le tiercé est affiché en gros caractères bleus au-dessus de l'hippodrome et une action sur la touche [ENTREE] permet de faire le point sur les finances.

Les joueurs n'ayant plus d'argent sont écartés du lot des parieurs et se trouvent ainsi condamnés à regarder les autres gagner ou perdre. Lorsque tout le monde est ruiné, le programme s'arrête de lui-même.

Structure du programme

Sous-programme d'affichage de la page de présentation : lignes 2000 à 2130

L'écran prend un fond vert (2000) et le mot TIERCE apparaît en bleu clair (ligne 2020). Des chevaux sont ensuite affichés au milieu d'une piste bordée de bandes rouges et blanches. Le message de début classique est imprimé grâce à la ligne 2110.

Sous-programme d'initialisation des "fichiers" des joueurs : lignes 3000 à 3050.

Le sous-programme prend connaissance du nombre de joueurs (3000) puis déclare des tableaux dont les dimensions sont fonction de ce nombre (ligne 3010).

Les lignes 3020 à 3040 rentrent le nom des joueurs.

Sous-programme de paris : lignes 5000 à 5150

Ce sous-programme enregistre les mises et les combinaisons des joueurs. Le nom ZZZZ est attribué aux joueurs ruinés permettant ainsi de les reconnaître et de ne plus les faire jouer.

Lignes 5020 à 5050, rentrée et test de validation de la mise.

Lignes 5060 à 5150, enregistrent la combinaison et vérifient d'une part qu'il n'y ait pas deux chevaux identiques dans un même tiercé (ligne 5160), et d'autre part que des numéros non compris entre 1 et 17 ne soient donnés (ligne 5120).

Sous-programme de dessin
de l'hippodrome : lignes 1000 à 1240.

Les lignes 1050 à 1110 dessinent la piste.

Les lignes 1140 à 1160 alignent les chevaux sur la ligne de départ.

Les lignes 1170 à 1230 affichent les numéros des chevaux.

Le programme principal s'étend depuis la ligne 0 jusqu'à la ligne 440.

Les lignes 80 à 110 permettent d'obtenir une valeur de la fonction RND totalement imprévisible.

Les lignes 200 à 310 gèrent le déplacement des chevaux.

Les lignes 320 à 360 enregistrent l'ordre d'arrivée des chevaux.

Les lignes 370 à 400 affichent le tiercé dans l'ordre.

La ligne 440 renvoie au commencement (course suivante).

```
0 CLEAR,,3:DIM CV(17,2),AR(3)
10 CONSOLE0,24:SCREEN 2,0,0:CLS
20 DEFGR$(1)=32,48,59,54,254,126,84,68
30 DEFGR$(0)=16,62,51,50,254,126,84,130
40 DEFGR$(2)=8,24,55,50,254,126,84,40
50 GOSUB 2000:GOSUB 3000
60 GOSUB 5000:GOSUB 1000
70 COLOR 0,2:LOCATE 0,0,0
80 PRINT"APPUYER SUR ENTREE POUR LE DEPART"
90 FOR I=1 TO 32700
100 A=RND:IF INKEY$=CHR$(13)THEN GOTO120
110 NEXTI:GOTO 80
120 LINE(0,0)-(39,0)CHR$(127),0
130 FOR I=1 TO 17
140 CV(I,1)=RND
150 CV(I,2)=2.
160 NEXT I
170 K=1
200 FOR I=1 TO 17
210 IF CV(I,1)>RND THEN GOTO 300
220 CV(I,2)=CV(I,2)+1
230 IF CV(I,2)=39 THEN GOTO 320
240 PSET(CV(I,2),4+I)GR$(INT(3*RND)),0
250 PSET(CV(I,2)-1,4+I)CHR$(127),2
300 NEXT I
310 GOTO 200
320 IF CV(I,2)=40 THEN GOTO 300
330 AR(K)=I:CV(I,2)=40:K=K+1
340 IF K<4 THEN GOTO 300
350 PSET(CV(I,2),4+I)GR$(INT(3*RND)),0
360 PSET(CV(I,2)-1,4+I)CHR$(127),2
370 ATTRB 1,1:COLOR 4,0
380 FOR I=1 TO 3
390 LOCATE 1+I*8,1,0:PRINT AR(I),:NEXT I
400 ATTRB 0,0:LINE(0,24)-(39,24)CHR$(127),0
410 COLOR2,0:CONSOLE 23,24:CLS
420 INPUT"APPUYER SUR ENTREE POUR AVOIR LES GAINS"
,A$
```

```
430 CONSOLE 0,24:CLS:GOSUB 4000
440 GOTO 60
1000 'HIP
1010 SCREEN,0,0:CLS
1050 CONSOLE 4,24:SCREEN 2,2:CLS
1060 LINE(0,2)-(39,2)CHR$(127),1
1070 LINE(0,3)-(39,3)CHR$(127),7
1080 LINE(0,23)-(39,23)CHR$(127),7
1090 LINE(0,24)-(39,24)CHR$(127),1
1100 LINE(28,20)-(28,192),1
1110 LINE(307,20)-(307,192),1
1140 FOR I=1 TO 17
1150 PSET(2,I+4)GR$(1),0,2
1160 NEXT I
1170 FOR I=1 TO 9
1180 PSET(0,I+4)CHR$(48+I),0,2
1190 NEXT I
1200 FOR I=0 TO 7
1210 PSET(1,I+14)CHR$(48+I),0,2
1220 PSET(0,I+14)CHR$(49),0,2
1230 NEXT I
1240 RETURN
2000 SCREEN ,2,2:CLS
2010 ATTRB 1,1:COLOR 6:LOCATE 13,5,0
2020 PRINT"TIERCE":COLOR 0
2030 FOR I=0 TO 2
2040 LOCATE 5+I*5,18-I*3
2050 PRINT GR$(I):NEXT I
2060 LINE(0,19)-(39,19)CHR$(127),7
2070 LINE(0,20)-(39,20)CHR$(127),1
2080 LINE(0,10)-(39,10)CHR$(127),7
2090 LINE(0,9)-(39,9)CHR$(127),1
2100 ATTRB 0,0:LOCATE 0,23:COLOR 0
2110 PRINT"APPUYER SUR ENTREE POUR COMMENCER"
2120 IF INKEY$<>CHR$(13)THEN GOTO 2080
2130 RETURN
3000 CLS:INPUT"COMBIEN Y A-T-IL DE PARIEURS";P
3010 DIM A$(P),SOM(P,5)
3020 FOR I=1 TO P
3030 PRINT"NOM DU JOUEUR ";I:INPUT A$(I)
3040 SOM(I,1)=1000:NEXT I
3050 RETURN
4000 FOR I=1 TO P
4010 TE=0:IF A$(I)="ZZZZ" THEN GOTO 4210
4020 FOR J=1 TO 3
4030 FOR K=1 TO 3
4040 IF AR(J)=SOM(I,K+2)THEN TE=TE+1
4050 NEXT K:NEXT J
4060 IF TE<>3 THEN GOTO 4200
4100 FOR J=1 TO 3
4110 IF AR(J)<>SOM(I,J+2) THEN GOTO 4150
```

```
4120 NEXT J
4130 PRINT A$(1);" A GAGNE DANS L'ORDRE",CHR$(13)
4140 SOM(I,1)=SOM(I,1)+SOM(I,2)*100:GOTO 4200
4150 PRINT A$(I);" A GAGNE DANS LE DESORDRE";CHR$(
13)
4160 SOM(I,1)=SOM(I,1)+SOM(I,2)*20
4200 IF SOM(I,1)=0 THEN PRINT A$(I);" EST RUINE IL
 NE JOUE PLUS":A$(I)="ZZZZ"
4210 NEXT I
4220 FOR I=1 TO P
4230 IF A$(1)<>"ZZZZ" THEN GOTO 4300
4240 NEXT I
4250 CLS:PRINT"TOUT LE MONDE ETANT RUINE, IL NE SE
MBLE PLUS UTILE DE CONTINUER":END
4300 FOR I=1 TO P:IF A$(1)="ZZZZ"THEN GOTO 4320
4310 PRINT A$(I);" POSSEDE: ";SOM(I,1)
4320 NEXT I
4330 INPUT"VOULEZ VOUS CONTINUER(O/N)";REP$
4340 IF LEFT$(REP$,1)="N" THEN CONSOLE 0,24:SCREEN
 2,0,0:END ELSE RETURN
5000 SCREEN 0,2,2:CLS
5010 FOR I=1 TO P:IF A$(I)="ZZZZ"THEN GOTO 5150
5020 PRINT"COMBIEN MISE ";A$(I);" ?"
5030 INPUT MISE:MISE=INT(ABS(MISE))
5040 S=SOM(I,1)-MISE:SOM(I,2)=MISE
5050 IF S<0 THEN PRINT"MISE TROP IMPORTANTE":GOTO
5020 ELSE SOM(I,1)=S
5060 PRINT "LE TIERCE DE ";A$(I);" EST:",CHR$(10),
CHR$(13)
5070 INPUT"PREMIER CHEVAL";SOM(1,3)
5080 INPUT"DEUXIEME CHEVAL";SOM(I,4)
5090 INPUT"TROISIEME CHEVAL";SOM(I,5)
5100 PRINT CHR$(10)
5110 FOR J=3 TO 5
5120 IF SOM(I,J)>17 OR SOM(I,J)<1 THEN PRINT"NUMER
O INTERDIT":GOTO 5060
5130 NEXT J
5140 IF SOM(1,3)=SOM(I,4) OR SOM(I,4)=SOM(I,5) OR
SOM(I,3)=SOM(I,5) THEN PRINT"INCORRECT: MEME NUMER
O PLUSIEUR FOIS":GOTO 5060
5150 NEXT I:RETURN
```

CHAPITRE **3**

Jeux de réflexion

3.1 CAVALIER

Les échecs fournissent un nombre colossal de problèmes qui sont indépendants du jeu lui-même. Beaucoup de problèmes utilisent le déplacement en L du cavalier. Il s'agit pour le joueur, en partant d'un endroit quelconque de l'échiquier, de déplacer le cavalier de telle manière que celui-ci passe une fois et une seule par toutes les cases de l'échiquier.

Le programme qui vous est proposé vous permet d'une part d'essayer vos facultés de réflexion, et d'autre part si vous pensez que la solution n'existe pas, de résoudre entièrement le problème grâce à notre TO7 qui, lui, est capable de démontrer le contraire.

Fonctionnement du programme

Après avoir effectué un RUN, la page de présentation du jeu apparaît. Vous devez alors choisir entre une démonstration (en appuyant sur la touche [D] puis [ENTREE]) et le jeu proprement dit (touche [J]).

Si vous choisissez la démonstration, vous n'aurez qu'à donner à votre TO7 la case de départ (la lettre puis le chiffre), le programme se chargera ensuite de remplir l'échiquier.

Si vous optez pour le jeu, il vous faudra alors indiquer au fur et à mesure les cases choisies (une question à-propos vous y invitera). Dans le cas fort probable où vous vous retrouveriez bloqué au bout d'un certain nombre de coups, vous aurez la possibilité de revenir en arrière d'autant de coups que vous le désirez ou d'abandonner. A tout moment, sauf au premier coup, vous avez également la possibilité de revenir en arrière ou d'abandonner ; pour cela, au lieu d'indiquer la "case suivante" vous mettrez "STOP" puis [ENTREE].

N'oubliez pas qu'il faut appuyer sur la touche [ENTREE] après avoir entré vos cases ou vos options. Pour les options, recopiez intégralement le texte indiqué entre parenthèses par votre TO7.

Structure du programme

Bloc de présentation : lignes 0 à 150

Les lignes 0 à 150 forment le bloc d'affichage de la page de présentation et aiguillent vers les lignes 1030 ou 2000 selon l'option demandée par l'utilisateur (jeu ou démonstration). On constate à la ligne 103 qu'en fait, si la première lettre de la réponse (D ou J) n'est pas un D, le programme considère qu'il s'agit d'un J.

Bloc d'initialisation de l'image et des tableaux : lignes 500 à 820

Ce bloc est appelé par les deux blocs qui suivent, il doit donc être utilisé comme sous-programme, ce qui impose l'utilisation d'un RETURN à la dernière ligne (820).

Les lignes 500 à 560 tracent le quadrillage délimitant les cases.

Les lignes 600 à 630 inscrivent la rangée de lettres et la colonne de chiffres.

Les lignes 670 à 690 permettent d'entrer la case de départ, de vérifier sa validité et d'afficher éventuellement un message d'erreur.

Les lignes 700 à 820 forment le sous-bloc d'initialisation du tableau des déplacements autorisés (8) selon les cases X et Y (770-810), et du tableau dans lequel se trouve la séquence des coups effectués par le joueur ou la machine (770-810). Ce dernier tableau 8 x 8 a un double rôle ; il permet d'une part de mémoriser les successions des coups du joueur afin de permettre le retour en arrière, et d'autre part de vérifier qu'une case donnée n'a pas déjà été utilisée. Le numéro du coup est inscrit dans la case du tableau correspondant à la case de l'échiquier. Les deux tableaux ont pour dimensions respectives (8, 2) et (8, 8) ; les indices 0 étant utilisés, les arguments de la ligne 10 sont (7, 1) et (7, 7).

Bloc de résolution par la machine : lignes 1000 à 1460

Ce bloc réalise principalement la résolution du problème de façon animée.

Nous n'entrerons pas dans les détails de mise en oeuvre de l'algorithme, signalons simplement qu'il s'agit de déplacer le cheval vers la case lui offrant le moins de possibilités de sortie. La méthode utilisée est celle trouvée par J.C WARNSDORFF en 1823.

Les lignes 1010 à 1020 et 1420 à 1460 permettent de poursuivre le déroulement du programme selon la volonté de l'utilisateur.

Bloc "joueur" : lignes 2000 à 4260

Cette partie du programme est destinée à faire jouer l'opérateur, tant en vérifiant la validité de ses déplacements que sa possibilité de poursuivre le jeu (blocage). En outre, les possibilités de retour en arrière et d'abandon ont été ajoutées.

Les lignes 2050 à 2110 détectent un déplacement non autorisé et le message d'erreur est affiché (ligne 4000). Si la case demandée est correcte, le déplacement s'effectue (2820 et 2830) ; la possibilité de continuer est testée par les lignes 2870 à 3020 et dans le cas où le joueur se trouve immobilisé, un message l'invitant à revenir en arrière ou à continuer apparaît (4005). Si les 64 cases sont effectivement parcourues, le message de la ligne 4100 est affiché puis le programme arrêté. La possibilité d'abandon ou de retour en arrière est assurée par la ligne 2040 à tout moment de la partie. Le retour en arrière est effectué par les lignes 4120 à 4260. Le principe en est simple, on cherche dans le tableau ECH le coup précédent puis on le supprime en enlevant le cheval de l'écran et en mettant à zéro la case correspondante de ECH.

```
0 'CAVALIER
10 CLEAR,,1:DIM DEP%(7,1),ECH%(7,7)
20 DEFINT A-Z
30 CONSOLE 0,24:SCREEN 0,3,3:CLS
40 DEFGR$(0)=8,28,126,126,14,28,62,126
50 ATTRB 1,1
60 FOR I=0 TO 9
70 LOCATE I*4,4:COLOR I MOD 2:PRINT GR$(0)
80 NEXT I
90 LOCATE 13,8:PRINT "JEU DU"
100 LOCATE 11,14:PRINT "CAVALIER"
110 ATTRB 0,0
120 COLOR 0:LOCATE 0,22:PRINT"VOULEZ VOUS UNE DEMO
NSTRATION (D)":INPUT"                          OU JOUER
 (J)",REP$
130 IF REP$="D" THEN GOTO 1030
140 GOSUB 500
150 GOTO 2000
500 'DEF DU JEU
510 CLS
520 FOR I=0 TO 16 STEP 2
530 LINE((12+I)*8+4,20)-((12+I)*8+4,148),0
550 LINE(100,(2+I)*8+4)-(228,(2+I)*8+4),0
560 NEXT I
600 FOR I=0 TO 14 STEP 2
610 PSET(13+I,1)CHR$(65+I/2),0
620 PSET(10,3+I)CHR$(48+I/2),0
630 NEXT I
670 LOCATE 0,22:INPUT"DONNEZ LA PREMIERE CASE:  ",
R$
680 V=ASC(RIGHT$(R$,1))-48:H=ASC(LEFT$(R$,1))-65
690 IF(H<0)OR(H>7)OR(V<0)OR(V>7)THEN LOCATE 0,22:I
NPUT"CASE INTERDITE; REJOUEZ ",R$:GOTO 680
700 DATA 2,1,2,-1,-2,1,-2,-1,1,2,1,-2,-1,2,-1,-2
710 RESTORE
720 FOR I=0 TO 7
730 FOR J=0 TO 1
740 READ DEP(I,J)
750 NEXT J
760 NEXT I
770 FOR I=0 TO 7
780 FOR J=0 TO 7
790 ECH(I,J)=0
800 NEXT J
810 NEXT I
820 RETURN
1000 'RESOLUTION
1010 CONSOLE 0,24:COLOR 0:CLS:LOCATE 0,22:INPUT"VO
ULEZ VOUS UNE PETITE DEMONSTRATION   ";REP$
1020 IF LEFT$(REP$,1)<> "O" THEN END
1030 GOSUB 500
1185 LOCATE 0,0,0
1190 PSET(H*2+13,V*2+3)GR$(0),1:PLAY"O5DO"
1200 ECH(H,V)=1:HP=H:VP=V
1210 FOR COUP=2 TO 64
```

```
1220 MIN=8:M=0
1230 FOR K=0 TO 7
1240 PCH=H+DEP(K,0):PCV=V+DEP(K,1)
1250 IF (0>PCH)OR(PCH>7)OR(0>PCV)OR(PCV>7)THEN GOT
O 1320
1255 IF ECH(PCH,PCV)<>0 THEN GOTO 1320
1260 N=0
1270 FOR L=0 TO 7
1280 H1=PCH+DEP(L,0):V1=PCV+DEP(L,1)
1290 IF (-1<H1)AND(H1<8)AND(-1<V1)AND(V1<8)THEN IF
 ECH(H1,V1)=0 THEN N=N+1
1300 NEXT L
1310 IF N<=MIN THEN MIN=N:DEPMIN=K
1320 NEXT K
1340 HP=H:VP=V:H=H+DEP(DEPMIN,0):V=V+DEP(DEPMIN,1)
1350 PSET(HP*2+13,VP*2+3)GR$(0),0:PSET(H*2+13,V*2+
3)GR$(0),1:PLAY "O5DO"
1360 ECH(H,V)=COUP:NEXT COUP
1420 LOCATE 0,22:PRINT "ET VOILA LE TRAVAIL !

1430 LOCATE 0,23:INPUT"VOULEZ VOUS ESSAYER";REP$:C
LS
1450 IF LEFT$(REP$,1)="O" THEN GOTO 140
1460 END
2000 'JOUEUR
2010 HP=H:VP=V:COUP=1:CONSOLE 22,23,,2
2020 PSET(H*2+13,V*2+3)GR$(0),1
2030 ECH(H,V)=COUP
2035 COLOR 0:LOCATE 5,0:PRINT "-----VOUS AVEZ JOUE
 ";COUP;" FOIS-----"
2040 COLOR 0:CLS:LOCATE 0,22:INPUT"CASE SUIVANTE "
;R$:IF LEFT$(R$,1)="S" THEN GOTO 4005
2045 V=ASC(RIGHT$(R$,1))-48:H=ASC(LEFT$(R$,1))-65
2050 IF(0>H)OR(H>7)OR(0>V)OR(V>7) THEN GOTO 4000
2060 IF ECH(H,V)<>0 THEN GOTO 4000
2070 IND=0
2080 FOR I=0 TO 7
2090 IF H-HP=DEP(I,0)THEN IF V-VP=DEP(I,1) THEN IN
D=1
2100 NEXT I
2110 IF IND=0 THEN GOTO 4000
2800 COUP=COUP+1:COLOR 0:LOCATE 5,0:PRINT "----VOU
S AVEZ JOUE ";COUP;" FOIS----"
2810 LOCATE 0,0,0
2820 PSET(HP*2+13,VP*2+3)GR$(0),0
2830 PSET(H*2+13,V*2+3)GR$(0),1
2870 N=0
2880 FOR I=0 TO 7
2890 H1=H+DEP(I,0):V1=V+DEP(I,1)
```

```
3000 IF (-1<H1)AND(H1<8)AND(-1<V1)AND(V1<8)THEN IF
 ECH(H1,V1)=0 THEN N=N+1
3010 NEXT I
3020 IF N=0 THEN GOTO 4100
3030 HP=H:VP=V
3040 GOTO 2030
4000 CLS:PLAY"O5DOREDO":LOCATE 0,22:INPUT "CASE IN
TERDITE ;REJOUEZ ",R$:GOTO 2045
4005 CLS:LOCATE 0,22:INPUT"VOULEZ VOUS REVENIR EN
ARRIERE(AR.)OU ABANDONNER(AB.)";REP$:GOTO 4110
4100 IF COUP=64 THEN LOCATE15,20:ATTRB 1,1:PLAY"O5
SISOL96LAL24":COLOR 1:PRINT"BRAVO":ATTRB 0,0:LOCAT
E 0,24:PRINT"JE N'AURAIS PAS FAIT MIEUX ! ":CONSOL
E 0,24,,0:COLOR0:END
4105 CLS:PLAY"O5SILASOLFAMIREDO":LOCATE 0,22:INPUT"
VOUS ETES BLOQUES;VOULEZ VOUS REVENIR EN ARRIERE(A
R.)OU ABANDONNER(AB.)",REP$
4110 IF REP$="AB."THEN GOTO 1000
4120 COLOR 0:CLS:LOCATE 0,22,0:INPUT"DE COMBIEN DE
 COUPS ";NC:LOCATE 0,0,0
4130 FOR C=NC TO 1 STEP-1
4140 ECH(H,V)=0:PSET(H*2+13,V*2+3)" ",3
4150 COUP=COUP-1
4160 FOR A=0 TO 7
4170 FOR B=0 TO 7
4220 IF ECH(A,B)=COUP THEN H=A:V=B
4230 NEXT B
4240 NEXT A
4250 NEXT C
4260 HP=H:VP=V:GOTO 2020
```

3.2 DAMIER SOLITAIRE

48 pions sont placés sur un damier de la façon suivante :

0 1 2 3 4 5 6 7

0 1 2 3 4 5 6 7

Le but du jeu est d'enlever le plus possible de pions en les prenant par des sauts en diagonale (comme au jeu de dames normal).

Pour prendre un pion, entrez, à la question "MOUVEMENT ?", les coordonnées de la case de départ et les coordonnées de la case d'arrivée sous forme d'un nombre de quatre chiffres. Pressez ensuite [ENTREE]. Le TO7 effectue la prise demandée, si le mouvement était possible, sinon un autre coup vous est demandé : chaque coup doit permettre la prise d'un pion !

Exemples :

MOUVEMENT ? 0022

MOUVEMENT ? 0033

MOUVEMENT INTERDIT, REJOUEZ

MOUVEMENT ? 2042

Si le nombre que vous donnez comporte un 8, la partie en cours s'arrête et le TO7 vous propose de rejouer. Cela vous permet d'arrêter le jeu quand il ne vous est plus possible de prendre.

Pour rejouer, pressez soit la touche [0] soit simplement la touche [ENTREE]. La touche [N] arrête définitivement le jeu.

DAMIER SOLITAIRE est un jeu très difficile. Enlever entre 40 et 45 pions est déjà une belle performance, en enlever 46 ou 47 un exploit ! Alors bon courage...

Structure du programme

Programme principal : lignes 5 à 1530

Ligne 5, réservation de place mémoire pour un caractère défini.

Ligne 10, appel du sous-programme de présentation.

Ligne 20, appel du sous-programme d'initialisation.

Ligne 30, entrée du mouvement - A$ est une variable chaîne contenant une trentaine d'espaces, ce qui permet d'effacer la quasi-totalité d'une ligne avant d'écrire dessus.

Lignes 40 à 60, rangement dans le tableau des quatre coordonnées données à la ligne 30.

Lignes 70 à 110, refus de toute coordonnée négative ou supérieure à 8 (ligne 80) - renvoi à la fin de partie si un chiffre entré est 8.

Ligne 110, mouvement interdit si la case de départ ne contient pas de pion (T(A(1), A(2)) = 0) ou si la case d'arrivée contient un pion (T(A(3), A(4)) = 1).

Ligne 120, refus de toute tentative de prise non conforme aux règles du jeu de dames.

Ligne 130, mouvement interdit si le saut se fait par-dessus une case vide et ne prend donc pas de pion.

Ligne 140, le pion pris est enlevé du terrain de jeu.

Ligne 150, le pion de la case de départ est transféré dans la case d'arrivée.

Ligne 160, le compteur de pions restants sur le damier est décrémenté d'une unité - appel du sous-programme d'affichage.

Ligne 170, au cas où un miracle vous aurait permis d'en arriver à n'avoir plus qu'un seul pion sur le damier, vous avez gagné !!

Ligne 180, retour à la ligne 30 pour une nouvelle prise.

Ligne 500, boucle de temporisation.

Ligne 520, effacement du message de mouvement interdit.

Ligne 530, retour en 30 pour une nouvelle prise.

Lignes 1000 et 1010, affichage du message de victoire (il faut tout prévoir, n'est-ce pas ...).

Lignes 1500 à 1530, fin de partie classique.

Sous-programme d'affichage : lignes 2000 à 2040

Ligne 2000, met un pion dans la case d'arrivée.

Ligne 2010, enlève le pion de la case de départ.

Ligne 2020, enlève le pion de la case par-dessus laquelle on a sauté.

Ligne 2030, affichage du nombre de pions restants sur le damier.

Ligne 2040, retour au programme principal.

Sous-programme d'initialisation :

Ligne 3000, effacement de l'écran.

Ligne 3010, compteur de pions à 48 et la chaîne A$ contient une trentaine d'espaces. Le tableau T représentant le damier est mis à zéro.

Lignes 3020 à 3060, il n'y a pas de pions au centre du damier au début du jeu (cf dessin dans la présentation du jeu). Un élément du tableau T à zéro correspond donc à un pion alors que cet élément vaut un si la case correspondante est vide.

Lignes 3070 à 3100, dessin du damier.

Lignes 3110 à 3140, affichage des repères autour du damier.

Lignes 3150 à 3190, on met un pion dans chaque case du damier.

Lignes 3200 à 3240, on enlève les pions du centre.

Ligne 3250, affichage du nombre initial de pions sur le damier.

Ligne 3260, retour au programme principal.

Sous-programme de présentation : lignes 3500 à 3620

Ligne 3500, effacement de l'écran - couleur de l'écran : jaune - couleur des caractères : noir.

Ligne 3510, déclaration des tableaux A (mouvements) et T (damier).

Ligne 3520, caractères en double largeur et double hauteur.

Ligne 3130, affichage du nom du jeu en rouge.

Ligne 3140, on revient à la couleur noire pour les caractères.

Ligne 3150, définition du caractère représentant un pion.

Lignes 3560 à 3600, un pion clignote à cinq reprises sous le nom du jeu.

Ligne 3610, attend que la touche [ENTREE] soit pressée pour laisser commencer le jeu.

Ligne 3620, retour au programme principal.

```
5 CLEAR,,1
10 GOSUB3500
20 GOSUB3000
30 LOCATE0,20,0:PRINT A$:LOCATE0,20,0:INPUT"MOUVEM
ENT";MOV$
40 FORI=1 TO 4
50 A(I)=VAL(MID$(MOV$,I,1))
60 NEXTI
70 FORI=1 TO 4
80 IF A(I)*(A(I)-8)>0 THEN 500
90 IF A(I)=8 THEN 1500
100 NEXTI
110 IF T(A(1),A(2))<>0 OR T(A(3),A(4))=0 THEN 500
120 IF ABS(A(1)-A(3))<>2 OR ABS(A(2)-A(4))<>2 THEN
 500
130 IF T(0.5*(A(1)+A(3)),0.5*(A(2)+A(4)))<>0 THEN
500
140 T(0.5*(A(1)+A(3)),0.5*(A(2)+A(4)))=1
150 T(A(1),A(2))=1:T(A(3),A(4))=0
160 H=H-1:GOSUB 2000
170 IF H=1 THEN 1000
180 GOTO 30
500 LOCATE0,22,0:PRINT"MOUVEMENT INTERDIT, REJOUEZ
":PLAY"A005T2L12DOMI"
510 FORI=1 TO 500:NEXTI
520 LOCATE0,22,0:PRINT A$
530 GOTO 30
1000 LOCATE0,0,0:PRINT A$
1010 LOCATE0,0,0:PRINT"BRAVO, VOUS AVEZ GAGNE !!!"
```

```
1500 LOCATE0,24,0:INPUT"VOULEZ-VOUS REJOUER";REP$
1510 IF LEFT$(REP$,1)="O" OR REP$="" THEN 20
1520 IF LEFT$(REP$,1)="N" THEN CLS:END
1530 GOTO 1500
2000 PSET(A(4)*2+13,A(3)*2+3)GR$(0),0
2010 PSET(A(2)*2+13,A(1)*2+3)" "
2020 PSET(A(2)+A(4)+13,A(1)+A(3)+3)" "
2030 LOCATE0,0,0:PRINT"PIONS:";H
2040 RETURN
3000 CLS
3010 H=48:A$="                                        ":FORI=0
TO8:FORJ=0TO8:T(I,J)=0:NEXTJ:NEXTI
3020 FORI=2TO5
3030 FORJ=2TO5
3040 T(I,J)=1
3050 NEXTJ
3060 NEXTI
3070 FORI=0 TO 16 STEP 2
3080 LINE((12+I)*8+4,20)-((12+I)*8+4,148),0
3090 LINE(100,(2+I)*8+4)-(228,(2+I)*8+4),0
3100 NEXTI
3110 FORI=0 TO 14 STEP 2
3120 PSET(13+I,1)CHR$(48+I/2),0
3130 PSET(10,3+I)CHR$(48+I/2),0
3140 NEXTI
3150 FORI=0 TO 7
3160 FORJ=0 TO 7
3170 PSET(I*2+13,J*2+3)GR$(0),0
3180 NEXTJ
3190 NEXTI
3200 FORI=2 TO 5
3210 FORJ=2 TO 5
3220 PSET(I*2+13,J*2+3)" "
3230 NEXTJ
3240 NEXTI
3250 LOCATE0,0,0:PRINT"PIONS:";H
3260 RETURN
3500 CLS:SCREEN0,2,2
3510 DIMA(4),T(8,8)
3520 ATTRB1,1:LOCATE4,10,0
3530 COLOR1,2:PRINT"DAMIER SOLITAIRE"
3540 COLOR0,2
3550 DEFGR$(0)=60,126,255,255,255,255,126,60
3560 FORI=1 TO 5
3570 LOCATE17,15,0:PRINT " "
3580 FORJ=1 TO 100:NEXT J
3590 LOCATE17,15,0:PRINT GR$(0):PLAY"AOT204L12FA"
3600 NEXTI
3610 ATTRB0,0:LOCATE0,24,0:INPUT"PRESSEZ ENTREE PO
UR COMMENCER",Q$
3620 RETURN
```

3.3 REVERSE

Voici une version complète et évoluée d'un jeu que nous avions déjà donné à titre d'exemple dans le livre "Guide du TO7".

Le principe de ce jeu de réflexion est le suivant: les chiffres 1, 2, 3, 4, 5, 6, 7, 8 et 9 sont placés dans un ordre quelconque par le TO7. Il s'agit d'ordonner ces chiffres par ordre croissant à partir de la gauche.

Pour ce faire, vous devez demander au TO7 d'effectuer les inversions de votre choix.

Exemples :

```
*  COUP : 0        135792846
   ROTATION ? 4
   COUP : 1        753192846
   ROTATION ?

*  COUP : 0        234516789
   ROTATION ? 4
   COUP : 1        543216789
   ROTATION ? 5
   COUP : 2        123456789
   GAGNE EN DEUX COUPS
```

Les rotations supérieures à 9 sont refusées. Demander 0 comme rotation en cours de partie permet d'arrêter le jeu.

Au début du jeu un nombre vous est demandé pour initialiser le générateur de nombres aléatoires (cf chapitre 1). Chaque fois que vous changerez de semence vous aurez une combinaison de départ différente. Il est intéressant de jouer plusieurs fois avec la même combinaison de départ (en entrant chaque fois la

même semence) pour essayer chaque fois de l'ordonner plus rapidement que la fois précédente et mesurer ainsi vos progrès à ce jeu.

Pour rejouer, pressez soit la touche [O] soit la touche [ENTREE] seule. La touche [N] permet d'arrêter le jeu.

Structure du programme

Programme principal : lignes 10 à 250

Ligne 10, appel du sous-programme de présentation.

Ligne 15, mise à zéro du score et effacement de l'écran.

Lignes 20 à 60, choix de la combinaison de départ. L'algorithme utilisé est le plus rapide que nous ayons pu mettre au point en BASIC, en trouverez-vous un plus rapide ?

Ligne 70, appel du sous-programme d'affichage pour donner la combinaison de départ.

Ligne 80, demande de la rotation.

Ligne 90, renvoi à la fin de la partie si la rotation rentrée précédemment est à 0.

Ligne 100, toute rotation supérieure à 9 est refusée.

Lignes 110 à 150, la rotation est exécutée - utilisation d'une variable de travail B.

Ligne 160, incrémentation d'une unité du nombre de coups déjà joués.

Ligne 170, appel du sous-programme d'affichage pour donner la combinaison obtenue après rotation.

Lignes 180 à 200, comparaison chiffre à chiffre de la combinaison obtenue avec la combinaison gagnante (123456789). Dès qu'une différence apparaît, la ligne 190 renvoie à la ligne 80 pour un nouveau coup.

Ligne 210, affiche le message de victoire si la comparaison précédente n'a pas fait apparaître de différence entre la combinaison obtenue et la combinaison gagnante.

Lignes 220 à 250, fin de partie classique.

Sous-programme d'affichage : lignes 1000 à 1050

Ligne 1000, affiche le numéro du coup.

Lignes 1010 à 1030, affiche la combinaison (remarquer le point-virgule dans la ligne 1020 pour obtenir tous les chiffres de la combinaison sur une même ligne).

Ligne 1040, saut d'une ligne.

Ligne 1050, retour au programme principal.

Sous-progamme de présentation : lignes 2000 à 2190

Ligne 2000, effacement de l'écran - couleur de l'écran : noir - couleur des caractères : vert - dimensionnement du tableau destiné à contenir la combinaison.

Ligne 2005, initialisation de la variable chaîne P$ avec la gamme chromatique - choix de l'attaque, du tempo et de la durée des notes.

Ligne 2010, positionnement du curseur et passage aux caractères de double hauteur et de double largeur.

Lignes 2020 et 2030, affichage multicolore du nom du jeu : REVERSE

Ligne 2040, retour à la taille normale des caractères.

Lignes 2050 et 2060 : affichage de deux combinaisons à titre d'exemple, la deuxième combinaison se déduisant de la première par une rotation 4.

Ligne 2070, caractère double hauteur et double largeur.

Lignes 2080 à 2130, clignotement de la combinaison gagnante obtenue par rotation 3 à partir de la combinaison précédente. La ligne 2120 affiche la combinaison. La ligne 2100 l'efface. La combinaison reste donc affichée à la sortie de la boucle.

Lignes 2140 à 2180, initialisation classique du générateur de nombres aléatoires.

Ligne 2190, retour au programme principal.

```
0 'REVERSE
10 GOSUB2000
15 S=0:CLS
20 FORI=1TO9
30 U=INT(RND*9+1)
40 IF A(U)<>0 THEN 30
50 A(U)=I
60 NEXTI
70 GOSUB1000
80 COLOR7,0:PRINT:INPUT"ROTATION";ROT:ROT=INT(ABS(
ROT))
90 IF ROT=0 THEN CLS:GOTO220
100 IF ROT>9 THEN 80
```

```
110 FORI=1 TO INT(ROT/2)
120 B=A(I)
130 A(I)=A(ROT-I+1)
140 A(ROT-I+1)=B
150 NEXTI
160 S=S+1
170 GOSUB1000
180 FORI=1 TO 9
190 IF A(I)<>I THEN 80
200 NEXTI
210 PRINT:PRINT"GAGNE EN ";S;" COUPS"
220 COLOR2,0:PRINT:PRINT:INPUT"VOULEZ-VOUS REJOUER
 ";REP$
230 IF LEFT$(REP$,1)="O" OR REP$="" THEN FORI=1 TO
9:A(I)=0:NEXTI:GOTO15
240 IF LEFT$(REP$,1)="N" THEN CLS:END
250 GOTO 220
1000 COLOR4,0:PRINT:PRINT"COUP:";:COLOR6,0:PRINT S
;"   ";:COLOR2,0
1010 FORI=1 TO 9
1020 PRINT A(I);
1030 NEXTI
1040 PRINT
1050 RETURN
2000 CLS:SCREEN2,0,0:DIM A(9)
2005 P$="DODO#RERE#MIFAFA#SOSO#LALA#SI":PLAY"AOT1L
4"
2010 LOCATE12,5:ATTRB1,1
2020 COLOR1,0:PRINT"R";:COLOR2,0:PRINT"E";:COLOR3,
0:PRINT"V";:COLOR4,0:PRINT"E";
2030 COLOR5,0:PRINT"R";:COLOR6,0:PRINT"S";:COLOR7,
0:PRINT"E"
2040 ATTRB0,0
2050 LOCATE11,10:COLOR2,0:PRINT"4 1 2 3 5 6 7 8 9"
2060 LOCATE11,12:COLOR4,0:PRINT"3 2 1 4 5 6 7 8 9"
2070 ATTRB1,1
2080 FORI=1 TO 5
2090 LOCATE2,15,0:COLOR1,0
2100 LOCATE2,15,0:PRINT"
2110 FORJ=1 TO 10:PLAY P$:NEXTJ
2120 LOCATE2,15,0:PRINT"1 2 3 4 5 6 7 8 9"
2130 NEXTI
2140 ATTRB0,0:LOCATE0,24,0:COLOR2,0:INPUT"ENTREZ U
N NOMBRE QUELCONQUE ";N
2150 N=INT(ABS(N))
2160 FORI=1 TO N
2170 Y=RND
2180 NEXTI
2190 RETURN
```

3.4 SIMON

Ce jeu, adaptation d'un jeu devenu maintenant célèbre, fait appel à votre mémoire visuelle.

Des lettres, choisies au hasard parmi les vingt-six lettres de l'alphabet, apparaissent les unes après les autres en des endroits quelconques de l'écran. Ces lettres forment ainsi une séquence dont vous devez être capable de redonner la composition exacte à la demande du TO7.

Chaque fois que vous avez réussi à donner toutes les lettres formant la séquence dans l'ordre exact de leur apparition, cette même séquence réapparaîtra augmentée d'une nouvelle lettre. le jeu se poursuit jusqu'à ce que vous finissiez par commettre une erreur ou un oubli.

Votre score est alors affiché ainsi que le meilleur score réalisé depuis le début du jeu.

Une nouvelle partie vous est ensuite proposée. Pressez soit la touche [O] soit [ENTREE] seulement pour rejouer. La touche [N] arrête le jeu définitivement.

Structure du programme

Programme principal : lignes 10 à 180

Ligne 10, appel du sous-programme de présentation.

Ligne 20, score et nombre de lettres de la séquence à zéro. Effacement de l'écran.

Ligne 30, choix de la première lettre.

Ligne 40, la séquence comporte une lettre de plus (au début du jeu C = 0 devient C = 1). Effacement de l'écran.

Lignes 50 à 120, affichage des lettres composant la séquence, chaque lettre (ligne 60) apparaissant fugitivement à une position donnée (lignes 70 et 80). L'intervalle de temps qui sépare l'apparition de deux lettres consécutives a une durée aléatoire (lignes 80 et 100).

Ligne 130, demande au joueur de redonner la séquence.

Ligne 140, compare la réponse du joueur et la réponse exacte - si la réponse est inexacte, renvoi en 1000 pour le message d'erreur.

Ligne 150, le score est augmenté de un - au cas (très improbable !!) où vous seriez arrivé à vous souvenir d'une séquence de 255 lettres (évidemment avec un papier et un crayon ...), le jeu s'arrête.

Ligne 160, une nouvelle lettre est ajoutée à la séquence.

Ligne 170, retour en 40 pour l'affichage de la séquence.

Ligne 1000, affichage d'un message d'erreur et de la bonne solution.

Ligne 1010, affichage du score.

Lignes 1030 et 1040, modification s'il y a lieu du meilleur score réalisé depuis le début du jeu.

Lignes 1050 à 1080, fin de partie classique.

Sous-programme de présentation : lignes 2000 à 2190

Ligne 2000, effacement de l'écran - couleur de l'écran : noir - couleur des caractères : vert - choix de l'attaque et du tempo des notes.

Lignes 2010 à 2120, boucle, parcourue dix fois au cours de laquelle une position, une couleur et une lettre sont choisies (lignes 2020 et 2030) ; la lettre est affichée avec un bip sonore (ligne 2040) ; le nom du jeu est affiché avec des couleurs qui changent sans cesse (lignes 2060 à 2110).

Ligne 2030, initialisation du record à zéro - déclaration des tableaux qui contiennent les coordonnées des positions où sont affichées les lettres composant la séquence.

Lignes 2140 à 2180, initialisation du générateur de nombres aléatoires.

Ligne 2190, retour au programme principal.

```
10 GOSUB2000
20 S=0:C=0:CLS
30 A$=CHR$(INT(RND*26+65))
40 C=C+1:CLS
50 FORI=1TOC
60 B$=MID$(A$,I,1)
70 X(C)=INT(RND*40):Y(C)=INT(RND*20)
80 LOCATEX(I),Y(I),0:PRINT B$:PLAY"L12DO"
90 D=INT(RND*500+100):FORJ=1TOD
100 NEXTJ
110 CLS
120 NEXTI
130 INPUT"REPONSE ";R$
140 IF R$<>A$ THEN 1000
150 S=S+1:IF S=255 THEN 1030
160 A$=A$+CHR$(INT(RND*26+65))
170 GOTO40
1000 LOCATE0,5,0:PRINT"ERREUR, LA BONNE REPONSE ET
AIT: ";A$
```

```
1010 LOCATE0,10,0:PRINT"VOUS ETES ALLE JUSQU""A ";
S;" LETTRES"
1020 LOCATE0,15
1030 IF S>RECORD THEN COLOR4,0:PRINT"NOUVEAU RECOR
D: ";S:RECORD=S:COLOR2,0:GOTO1050
1040 PRINT"RECORD TOUJOURS A: ";RECORD
1050 LOCATE0,20:INPUT"VOULEZ VOUS REJOUER ";REP$
1060 IF REP$="" OR LEFT$(REP$,1)="O" THEN 20
1070 IF LEFT$(REP$,1)="N" THEN CLS:END
1080 GOTO1050
2000 SCREEN2,0,0:CLS:PLAY"A0T4"
2010 FORI=1TO10
2020 X=INT(RND*40):Y=INT(RND*20)
2030 C=INT(RND*7+1):C$=CHR$(INT(RND*26+65))
2040 ATTRB0,0:LOCATEX,Y,0:COLORC,0:PRINT C$:PLAY"L
12DO"
2050 ATTRB1,1:LOCATE15,10,0
2060 COLOR1,0:PRINT"S";:COLOR2,0:PRINT"I";:COLOR3,
0:PRINT"M";
2070 COLOR4,0:PRINT"O";:COLOR5,0:PRINT"N"
2080 FORJ=1TO50:NEXTJ
2090 LOCATE15,10,0
2100 COLOR5,0:PRINT"S";:COLOR4,0:PRINT"I";:COLOR3,
0:PRINT"M";
2110 COLOR2,0:PRINT"O";:COLOR1,0:PRINT"N"
2120 NEXTI
2130 RECORD=0:DIM X(255),Y(255)
2140 ATTRB0,0:COLOR2,0:LOCATE0,24,0:INPUT"ENTREZ U
N NOMBRE QUELCONQUE ",N
2150 N=INT(ABS(N))
2160 FORI=1TON
2170 Y=RND
2180 NEXTI
2190 RETURN
```

3.5 SOLITAIRE

C'est un prisonnier de la Bastille qui au XVIIème siècle inventa ce jeu pour calmer sa solitude. A l'origine, le jeu était constitué d'une planchette octogonale percée de 37 trous dans lesquels s'enfonçaient les fiches. Pour jouer, on retire une fiche quelconque, puis on "mange" la fiche voisine d'un trou, en la sautant dans le sens horizontal ou vertical (donc, seulement si un trou se trouve derrière elle). Pour gagner, il faut parvenir à manger toutes les fiches de façon à ce qu'une seule fiche reste sur la tablette en fin de partie.

Deux siècles plus tard, la tablette antique et respectable a été remplacée par un écran vidéo aux usages multiples et les fiches par des caractères graphiques de forme circulaire. Cependant, les hommes n'ont pas changé et ont souvent beaucoup de mal à résoudre ce jeu de réflexion.

Si vous connaissez le jeu, vous apprécierez la souplesse d'utilisation qu'offre un ordinateur et dans le cas contraire, empressez-vous de combler cette lacune.

		x	x	x		
	x	x	x	x	x	
x	x	x	x	x	x	x
x	x	x	x	x	x	x
x	x	x	x	x	x	x
	x	x	x	x	x	
		x	x	x		

Fonctionnement du jeu

Après avoir effectué une commande RUN, l'habituelle page de présentation apparaît. Vous devez alors appuyer sur [ENTREE] comme l'indique le message au bas de l'écran. La table de jeu se dessine alors et les pions sont repérés par une rangée de lettres et une colonne de chiffres (0 à 6). une question vous invite à enlever le premier pion, pour cela indiquez en premier lieu la lettre puis le chiffre.

Exemple :

C2 [ENTREE]

Immédiatement après avoir enfoncé la touche [ENTREE], le pion désigné disparaît et c'est maintenant que le jeu de réflexion commence véritablement.

Pour manger, il vous faut tout d'abord indiquer le pion qui va se déplacer (et donc manger son voisin).

Exemple :

CO [ENTREE]

(En réponse à la question "quel pion voulez-vous déplacer ?").

Le pion ainsi désigné prend une couleur verte et la machine vous demande le sens du déplacement de ce pion. Le sens de déplacement sera indiqué à l'aide d'une des quatre touches :
← , → , ↑ , ↓ suivant le sens indiqué par la flèche, vous pouvez voir le pion vert manger son voisin.

Exemple : ↓

(il suffit d'enfoncer la touche pour provoquer le mouvement).

Après cette opération, le pion redevient rouge.

Dans le cas où une case serait incorrecte ou bien un déplacement illégal, un message vous invite à recommencer.

Si vous êtes bloqué, la machine vous l'indique et sort du programme ; elle saura également reconnaître votre victoire éventuelle.

Structure du programme

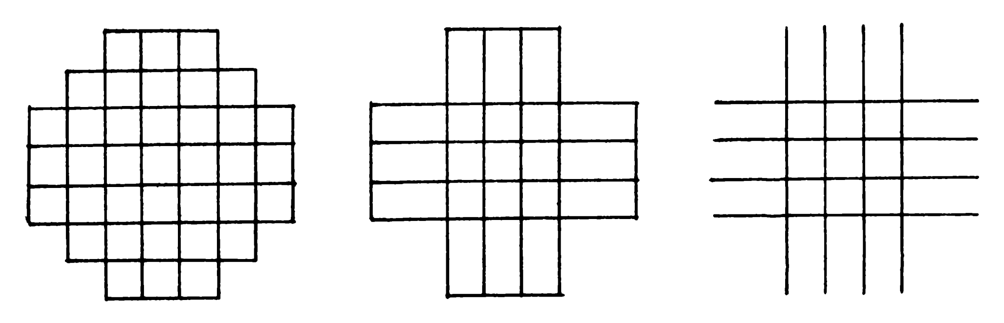
Les lignes 0 à 140 représentent une phase d'initialisation et appellent trois sous-programmes.

Sous-programme de présentation : lignes 4000 à 4130

Ces quelques lignes permettent d'afficher la page de présentation indispensable pour rendre un jeu attrayant et propre.

Sous-programme de dessin de la tablette

La géométrie de la table de jeu sort de l'ordinaire, aussi, un programme un peu plus long que d'habitude est utilisé. le quadrillage est dessiné en trois fois. Les phases successives sont représentées ci-dessous.



phase 1 phase 2 phase 3

Les lignes 1010 à 1040	effectuent la phase 1
Les lignes 1050 à 1080	effectuent la phase 2
La ligne 1090	effectue la phase 3

Sous-programme d'initialisation du tableau de jeu et de la distribution des pions

Le tableau JEU de dimensions 7 x 7 est tout d'abord initialisé à 1 (lignes 2010 à 2040), puis les cases interdites (trois par angle) sont mises à la valeur 0. Enfin, et grâce au tableau qui vient d'être

créé, les pions sont disposés sur la tablette (2060 à 2130). On remarquera que la rangée de lettres ainsi que la colonne de chiffres sont également inscrites grâce à ces deux boucles imbriquées.

Bloc du jeu proprement dit

Les lignes 150 à 490 constituent ce bloc. Les questions adéquates sont posées à l'utilisateur et les déplacements et tests correspondant aux réponses sont effectués.

Les lignes 150 à 170 correspondent à la question "Quel pion voulez-vous déplacer ?"

Les lignes 180 à 280 analysent le sens de déplacement (entré à l'aide d'une des quatre touches fléchées du clavier).

Les lignes 290 à 370 effectuent ce déplacement.

Les lignes 380 à 470 vérifient que le joueur n'est pas bloqué ; pour cela, il suffit de vérifier qu'au moins deux pions se trouvent sur des cases voisines.

Sous-programme d'entrée d'une case et de vérification de sa validité (900 à 940)

Ce sous-programme convertit la valeur chaîne entrée (exemple : C7) en deux nombres représentant les coordonnées de la case (exemple : C3, 7). Il vérifie ensuite que cette case est bien autorisée.

```
0 'SOLITAIRE
10 CLEAR,,1
20 CONSOLE 0,24:SCREEN 4,0,0:CLS
30 DIM JEU(6,6)
40 DEFGR$(0)=60,126,255,255,255,255,126,60
50 GOSUB 4000
60 GOSUB 1000
70 GOSUB 2000
100 CONSOLE 21,24:COLOR 2:LOCATE 0,22
110 INPUT"QUEL EST LE PREMIER PION A ENLEVER";A$
130 GOSUB 900:IF FLAG=1 THEN GOTO 110
140 JEU(X,Y)=2:PSET(13+X*2,4+Y*2)CHR$(127),0:CLS:C
OUP=1
150 COLOR 2:LOCATE 0,23:INPUT"QUEL PION VOULEZ VOU
S DEPLACER";A$
160 GOSUB 900:IF FLAG=1 THEN GOTO 150
165 IF JEU(X,Y)=2 THEN GOSUB 930:GOTO 150
170 PSET(13+X*2,4+Y*2)GR$(0),2
180 CLS:PRINT"SENS DE DEPLACEMENT ?"
190 A$=INKEY$:IF(A$="")THEN GOTO 190:IF (ASC(A$)<8
)OR(ASC(A$)>11)THEN GOTO 190
200 ON ASC(A$)-7 GOTO 210,220,230,240
210 Y1=Y:Y2=Y:X1=X-1:X2=X-2:GOTO 250
220 Y1=Y:Y2=Y:X1=X+1:X2=X+2:GOTO 250
230 X1=X:X2=X:Y1=Y+1:Y2=Y+2:GOTO 250
240 X1=X:X2=X:Y1=Y-1:Y2=Y-2
250 V=X:W=Y
260 X=X1:Y=Y1:GOSUB 910:IF FLAG=1 THEN GOTO 500
270 X=X2:Y=Y2:GOSUB 910:IF FLAG=1 THEN GOTO 500
280 IF(JEU(X1,Y1)<>1)OR(JEU(X2,Y2)<>2)THEN GOSUB 9
30:GOTO 500
285 LOCATE 0,0,0
290 PSET(13+V*2,4+W*2)CHR$(127),0
300 PSET(13+X1*2,4+Y1*2)GR$(0),2
310 FOR I=1 TO 200:NEXT I
320 PSET(13+X1*2,4+Y1*2)CHR$(127),0
330 PSET(13+X2*2,4+Y2*2)GR$(0),2
340 FOR I=1 TO 200:NEXT I
350 PSET(13+X2*2,4+Y2*2)GR$(0),1
360 JEU(V,W)=2:JEU(X1,Y1)=2:JEU(X2,Y2)=1
370 COUP=COUP+1:IF COUP=36 THEN GOTO 3000
380 FOR I=0 TO 6
390 FOR J=0 TO 6
400 IF JEU(I,J)<>1 THEN GOTO 450
410 IF I+1<7 THEN IF JEU(I+1,J)=1 GOTO 490
420 IF I-1>-1THEN IF JEU(I-1,J)=1 GOTO 490
430 IF J+1<7THEN IF JEU(I,J+1)=1 GOTO 490
440 IF J-1>-1THEN IF JEU(I,J-1)=1 GOTO 490
450 NEXT J:NEXT I
```

```
460 CLS:ATTRB1,1:LOCATE 5,24:PRINT"VOUS ETES BLOQU
E"
470 ATTRB 0,0:CONSOLE 0,24:COLOR2:LOCATE0,0:END
490 CLS:GOTO 150
500 PSET(13+V*2,4+W*2)GR$(0),1:GOTO150
900 X=ASC(LEFT$(A$,1))-65:Y=ASC(RIGHT$(A$,1))-48
910 FLAG=0:IF (X<0)OR(X>6)OR(Y<0)OR(Y>6)THEN GOTO
930
920 IF JEU(X,Y)<>0 THEN GOTO 940
930 FLAG=1:CLS:PRINT"CASE INCORECTE"
940 RETURN
990 END
1000 'DESSIN
1010 CLS:COLOR 4:FOR I=0 TO 3
1020 LINE(131+I*16,27)-(131+I*16,139)
1030 LINE(99,59+I*16)-(211,59+I*16)
1040 NEXT I
1050 FOR I=0 TO 112 STEP 112
1060 LINE(131,27+I)-(179,27+I)
1070 LINE(99+I,59)-(99+I,107)
1080 NEXT I
1090 BOX(115,43)-(195,123)
1100 RETURN
2000 'PIONS
2010 FOR I=0 TO 6
2020 FOR J=0 TO 6
2030 JEU(I,J)=1
2040 NEXT J:NEXT I
2050 JEU(0,0)=0:JEU(0,1)=0:JEU(1,0)=0:JEU(6,6)=0:J
EU(6,5)=0:JEU(5,6)=0:JEU(0,5)=0:JEU(0,6)=0:JEU(1,6
)=0:JEU(5,0)=0:JEU(6,0)=0:JEU(6,1)=0
2060 FOR I=0 TO 6
2070 FOR J=0 TO 6
2080 IF JEU(I,J)=0THEN GOTO 2100
2090 PSET(13+I*2,4+J*2)GR$(0),1
2100 NEXT J
2110 PSET(13+I*2,1)CHR$(65+I),2
2120 PSET(10,4+I*2)CHR$(48+I),2
2130 NEXT I
2140 RETURN
3000 'GAGNE
3010 ATTRB 1,1:COLOR 1:CLS:LOCATE 14,24:PRINT"BRAV
O":CONSOLE 0,24:COLOR 2:ATTRB0,0
3020 LOCATE0,22:END
4000 'PRESENTATION
4010 CLS:ATTRB 1,1:LOCATE 15,10
4020 COLOR 4:PRINT"SOLITAIRE"
4030 COLOR 1
4040 LOCATE 14,12:PRINT"=========="
4050 FOR I=0 TO 4
4060 LOCATE 0+I*5,12+I*2
```

```
4070 COLOR I+1:PRINT GR$(0)
4080 LOCATE 0+I*5,12-I*2:PRINT GR$(0)
4090 NEXT I
4100 ATTRB 0,0:LOCATE 0,23:COLOR 2
4110 PRINT"APPUYER SUR ENTREE POUR COMMENCER"
4120 IF INKEY$<>CHR$(13)THEN GOTO 4120
4130 RETURN
```

CHAPITRE **4**

Jeux d'action

4.1 ASTÉROÏDES

Grâce à votre TO7, vous allez participer à une phase décisive de la guerre des étoiles. Alors que les armées alliées ont réussi à installer de solides défenses pour empêcher toute intrusion de l'ennemi, voilà qu'un essaim d'astéroïdes est signalé dans le secteur où vous effectuez une reconnaissance sans arme. le piège est évident : Les forces ennemies vont profiter de cet essaim pour faire pénétrer leurs plus puissants vaisseaux de guerre au coeur du système de défense allié. Il vous faut donc au péril de votre vie en détruire le plus grand nombre ; n'étant pas armé, vous allez utiliser le fait que votre champ de force est le plus puissant pour faire exploser les lourds navires de guerre en les percutant.

Mais attention ! si votre champ de force peut aussi résister aux chocs latéraux avec les astéroïdes, il n'en est pas de même d'une collision frontale qui vous serait fatale.

Disposition du jeu

Votre appareil est de couleur rouge et se situe en haut de l'écran ; les astéroïdes magenta ainsi que les vaisseaux jaunes à abattre se déplacent vers le haut de l'écran. Le score est inscrit en vert sur la dernière ligne de l'écran. Le tout se présente sur fond noir. Lors d'un choc, une explosion est visualisée à la place du vaisseau détruit. Les touches ← et → permettent le déplacement.

Après avoir lancé le programme à l'aide de la commande RUN, la page de présentation apparaît. Il vous suffit alors d'appuyer sur la touche [ENTREE] pour commencer le combat. Pour rejouer, il faut répondre par [O] ou tout autre mot commençant par la lettre O puis actionner la touche [ENTREE].

Structure du programme

Les lignes 1 à 90 forment le bloc d'initialisation. Le sous-programme 1000 fait également partie de ce bloc puisqu'il y est appelé.

Les lignes 100 à 200 forment le coeur du programme, c'est dans ce bloc qu'est effectué le déplacement du vaisseau rouge (joueur), on y trouve également la génération aléatoire des astéroïdes ainsi que celle des vaisseaux et enfin le test de collision qui, s'il est vérifié, envoie aux lignes 600 à 760.

Les lignes 600 à 760 forment le bloc "COLLISION" où est analysée la nature de la collision, puis générée l'explosion sur l'écran. S'il s'avère que le vaisseau du joueur est détruit, alors le message indiquant la fin de la partie est affiché.

Analyse détaillée du programme

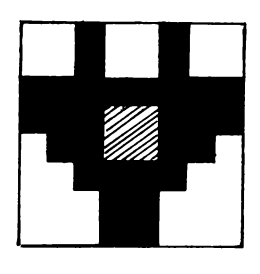
La ligne 1 représente l'initialisation indispensable à tout programme.

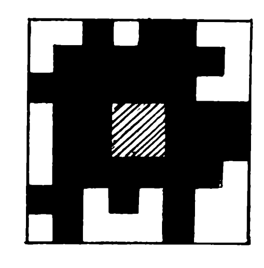
En effet, nous rappelons que-pour que l'affichage soit indépendant du passé de votre ordinateur,-vous devez positionner toutes les variables susceptibles d'avoir été modifiées antérieurement au démarrage du programme. C'est pourquoi une instruction comme : CONSOLE 0, 24 qui peut paraître inutile à première vue est en fait indispensable pour un déroulement sans aléas du programme.

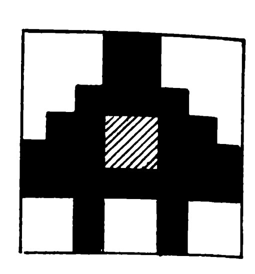
Les lignes 10 à 60 définissent les différents caractères graphiques dans l'ordre suivant :

graphique 0 : fusée jaune
graphique 1 : astéroïdes
graphique 2 : première phase de l'explosion
graphique 3 : deuxième phase de l'explosion
graphique 4 : troisième phase de l'explosion
graphique 5 : fusée ennemie

Il est important de signaler que les caractères graphiques susceptibles d'être testés par le programme (ici 0, 1 et 5) doivent obligatoirement avoir les quatre points centraux allumés sur l'écran (partie hachurée de la figure ci-dessous) pour être compatibles avec le programme qui vous est donné. Moyennant cette petite restriction, vous pouvez les modifier à votre gré. Ceux que nous vous proposons sont les suivants :


fusée joueur


astéroïdes


fusée ennemie

A la ligne 130, dans un premier temps, on mémorise la commande envoyée par le joueur et dans un deuxième temps, on réalise le déplacement correspondant sauf, bien sûr, si le joueur se trouve sur les limites de l'écran.

Prenons un exemple : la touche ← a été enfoncée.

La valeur logique de (IN$ = CHR$(8)) vaut -1 (-1 correspond à une expression vraie et 0 à une expression fausse) et celle de (IN$ = CHR$(9)) vaut 0 puisque IN$ représente la touche enfoncée et CHR$(8) représente la touche . D'autre part, si le joueur n'est pas déjà sur la colonne 0 (tout à gauche de l'écran), alors la valeur logique de (P > 0) vaut -1 ; donc l'expression

- (IN$ = CHR$(8)) * (P > 0) vaut -1

et l'expression

+ (IN$ = CHR$ (9)) * (P < 39) vaut 0

A la ligne 130, tout se passe comme si on avait P = P - 1 et la fusée affichée lors de l'exécution de la ligne 140 se sera déplacée d'un cran vers la gauche (P représente la position horizontale de la fusée du joueur).

La ligne 140 affiche la fusée du joueur avec la couleur rouge et selon la position calculée à la ligne 130.

Les lignes 150 à 170 placent trois astéroïdes par ligne sur l'écran et ceci de façon aléatoire (bien sûr, il peut y avoir deux astéroïdes au même endroit mais ceci n'a aucune importance).

La ligne 180 ajoute en plus, de façon aléatoire (une ligne sur dix en moyenne), une fusée ennemie de couleur jaune.

Il est à noter qu'en modifiant ces lignes, il est possible de rendre le jeu plus difficile (ou plus facile !) :

. soit en diminuant le nombre de fusées ennemies: en augmentant le nombre 0.9 dans IF RND 0.9 (il faut toujours que ce nombre soit inférieur à 1).

. soit en augmentant le nombre d'astéroïdes par ligne ; pour cela, il faudra ajouter autant de lignes identiques à 150 que d'astéroïdes. Il faut faire une remarque très importante à ce sujet. On aurait pu remplacer les lignes 140 à 170 par une boucle (FOR - NEXT) mais ceci se serait fait aux dépens de la rapidité du programme.

La ligne 190 teste si le caractère graphique situé immédiatement au-dessous de la fusée du joueur est d'une autre couleur que le rouge ou le noir, dans ce cas, on va à la ligne 600 qui traite la collision.

La ligne 200 donne le mouvement au jeu. En effet, le caractère CHR$(13) correspond à un passage à la ligne suivante. Or le curseur se trouve précisément à la dernière ligne de la fenêtre de travail, tout l'écran va donc se déplacer d'une ligne vers le haut ce qui donne cette impression de déplacement que vous pourrez constater après avoir entré le programme dans votre TO7. Après cette instruction, le programme retourne à la ligne 130 et tout recommence.

La ligne 610 met Q à 1 si la collision s'est faite avec un vaisseau ennemi et à 0 dans le cas d'un astéroïde.

Les lignes 620 à 660 créent la scène d'explosion là où se trouve le vaisseau ennemi en Q = 1 et à la place du vaisseau joueur en Q = 0.

Lignes 670 et 680, le score est calculé, affiché et en Q = 1 on peut continuer le programme en retournant à la ligne 200; dans le cas contraire, le message de fin de jeu est affiché et votre réponse à la question "VOULEZ-VOUS REJOUER ?" est analysée puis traitée (lignes 700 - 760). On remarquera l'utilisation des instructions SCREEN et CONSOLE de façon abondante. L'instruction CONSOLE permet d'écrire le message au bas de l'écran sans faire disparaitre ni "glisser" l'image lors d'un "CLS" et "INPUT" respectivement. L'instruction LEFT$ (ligne 750) permet de ne prendre en compte que la première lettre de la réponse ; ainsi, des réponses comme : "OUI", "OU", "OUI.", "O" ou même "OK" seront interprétées comme une réponse "OUI".

Le sous-programme 1000 sert uniquement à afficher la page de présentation (lorsque vous effectuez la commande RUN), il est très classique et le lecteur pourra comprendre son sens sans difficulté.

Maintenant que vous avez bien compris la structure de votre programme, vous pourrez le "rentrer" intelligemment dans votre TO7 en ayant moins de risques de vous tromper puisque vous aurez un esprit plus critique.

Alors bonne chance dans cette difficile bataille de l'espace !

```
0 'ASTEROIDE
1 CLEAR ,,6:CONSOLE 0,24:SCREEN 2,0,0:CLS
10 DEFGR$(0)=36,36,255,255,126,60,24,24
20 DEFGR$(1)=44,128,252,127,127,126,212,68
30 DEFGR$(2)=0,0,0,24,24,0,0,0
40 DEFGR$(3)=0,16,60,124,62,60,8,0
50 DEFGR$(4)=89,118,124,124,254,124,62,83
60 DEFGR$(5)=24,24,60,126,255,255,36,36
70 GOSUB 1000
80 SCREEN,0,0:CLS:CONSOLE 0,23
90 COLOR 2:LOCATE 5,24,0:PRINT"SCORE  0"
100 SCORE=0
110 P=19
120 PSET(P,0)GR$(0),1
130 IN$=INKEY$:P=P-(IN$=CHR$(8))*(P>0)+(IN$=CHR$(9
))*(P<39)
140 PSET(P,0)GR$(0),1
150 PSET(40*RND,23)GR$(1),5
160 PSET(40*RND,23)GR$(1),5
170 PSET(40*RND,23)GR$(1),5
180 IF RND>0.9 THEN PSET(40*RND,23)GR$(5),3
190 IF POINT(P*8+3,11)>1 THEN GOTO 600
200 LOCATE 22,24:PRINT CHR$(13):GOTO 130
600 'COLLISION
610 Q=-(POINT(P*8+3,11)=3)
620 PSET(P,Q)GR$(2),7
630 FOR I=1 TO 150:NEXT I
640 PSET(P,Q)GR$(3),3
650 FOR I=1 TO 100:NEXT I
660 PSET(P,Q)GR$(4),1
670 SCORE=SCORE+Q*47:COLOR 2:LOCATE 12,24:PRINT SC
ORE;CHR$(11)
680 IF Q=1 THEN GOTO 200
700 CONSOLE 22,24:CLS:LOCATE 0,22:COLOR 2
710 PRINT"VOTRE VAISSEAU EST DETRUIT"
720 PRINT "VOTRE SCORE EST DE ";SCORE;"POINTS"
730 INPUT"VOULEZ VOUS REJOUER";RE$
740 CONSOLE 0,24
```

```
750 IF LEFT$(RE$,1)="O"THEN GOTO 80
760 CLS:SCREEN 2,0,0:CLS:END
1000 ' PRESENTATION
1010 ATTRB 1,1:COLOR 5
1020 FOR I=0 TO 38 STEP 4
1030 FOR J=0 TO 20 STEP 4
1040 IF INT(3*RND)=1 THEN PSET(I,J)GR$(1),5
1050 NEXT J
1060 NEXT I
1100 LOCATE 10,8:COLOR 4
1110 PRINT "ASTEROIDES":LOCATE 8,10:COLOR 1:PRINT"
=============="
1120 LOCATE 22,16:PRINT GR$(0)
1130 COLOR 3:LOCATE 22,20:PRINT GR$(5)
1200 ATTRB 0,0:LOCATE 10,24:COLOR 2:INPUT"ETES VOU
S PRET";A$
1210 RETURN
```

4.2 BOMBARDIER

Prenez quelques instants les commandes d'un chasseur bombardier de l'Aéronavale et tentez en 50 passages de couler le maximum de bateaux de guerre en mouvement. L'épreuve est d'autant plus difficile que bateau et avion peuvent se déplacer dans les deux sens de façon totalement indépendante et aléatoire. Les bombes dont vous diposez ont leur propre mouvement et, l'avion passant à des altitudes toujours différentes, il vous faudra être très adroit.

Fonctionnement du jeu

En exécutant la commande RUN, la présentation représentant un avion en pleine attaque apparaît. Une action sur la touche [ENTREE] dessine l'océan vide de toute violence. Pendant ce temps, le compteur aléatoire tourne, assurant des séquences toujours différentes. Le comptage est interrompu par l'enfoncement de la touche [ENTREE], les 50 passages commencent alors.

L'avion pénètre sur l'écran par la gauche ou la droite de façon imprévisible et à une altitude toujours différente. Le bateau se déplace moins vite que l'avion mais dans un sens tout aussi imprévisible. le largage de la bombe s'effectue par une action sur la touche [F]. Le score est augmenté lorsque le bateau est touché (en fonction de la difficulté). Si le bateau se dirige en un sens inverse de l'avion, les points sont doubles et plus l'avion est haut, plus le fait de couler un bateau rapporte de points.

Structure du programme

Les lignes 10 à 70 définissent les caractères graphiques (avion dans les deux sens, bateau et bombe dans les deux sens). Le bateau est construit à l'aide de deux caractères graphiques (voir annexe).

Les lignes 240 à 370 gèrent le cas où l'avion se déplace de gauche à droite. Dans ce bloc, le lancement de la bombe est effectué (ligne 280) ainsi que son déplacement (ligne 290). Le déplacement du bateau se trouve sur la ligne 320. La ligne 330 calcule le score et les lignes 340 à 360 font couler le bateau.

Les lignes 400 à 520 gèrent l'autre possibilité de déplacement de l'avion. Ce bloc a la même structure que le précédent.

La ligne 210 a auparavant tiré de façon aléatoire l'altitude de l'avion, la position de départ du bateau ainsi que son sens de déplacement. Le sens de déplacement de l'avion est déterminé de façon aléatoire à la ligne 230 qui branche selon le cas en 240 ou en 400.

Les lignes 700 et 710 invitent le joueur à recommencer une partie ou à sortir du programme.

Les lignes 1000 à 1020 créent un rectangle bleu qui représente la mer.

Les lignes 2000 à 2120 forment le sous-programme de présentation ; les caractères graphiques y sont utilisés comme pour toute présentation personnalisée d'un programme de jeu.

```
10 CLEAR,,6
20 DEFGR$(0)=0,3,131,239,252,160,48,48
30 DEFGR$(1)=0,192,193,247,63,5,12,12
40 DEFGR$(2)=1,1,253,31,255,127,63,31
50 DEFGR$(3)=1,130,188,248,255,254,252,248
60 DEFGR$(4)=1,3,63,255,255,63,3,1
70 DEFGR$(5)=128,192,252,255,255,252,192,128
90 GOSUB 2000
100 SC=0
110 GOSUB 1000
120 COLOR 0,6:LOCATE 0,0,0:PRINT "APPUYER SUR ENTR
EE POUR COMMENCER"
130 A=RND
140 IF INKEY$<>CHR$(13)THEN GOTO 130
150 LOCATE 0,0,0:PRINT SPC(39)
160 LOCATE 1,0,0:PRINT"SCORE"
200 FOR K=1 TO 50
210 DB=SGN(RND-0.5):BA=14+INT(RND*10):V=2+INT(RND*
15)
220 PSET(BA,21)GR$(2),0:PSET(BA+1,21)GR$(3),0
230 IF RND>0.5 THEN GOTO 400
240 PSET(0,V)GR$(1),0
250 FOR I=1 TO 39
260 PSET(I-1,V)CHR$(127),6,6:PSET(I,V)GR$(1),0
270 IF FLAG=1 THEN GOTO 290
280 IF INKEY$="F" THEN FLAG=1:PSET(I,V+1)GR$(5),1:
B=V+1:GOTO 310ELSE GOTO 310
290 PSET(I-1,B)CHR$(127),6,6:B=B+1:PSET(I,B)GR$(5)
,1:IF B<21 THEN GOTO 310
300 IF I=BA OR I=BA+1 THEN GOTO 330 ELSE FLAG=0:PS
ET(I,21)CHR$(127),6,6
310 IF (I MOD3=0)THEN PSET(BA-(DB<0),21)CHR$(127),
6,6:BA=BA+DB:PSET(BA,21)GR$(2),0:PSET(BA+1,21)GR$(
3),0
320 NEXT I:FLAG=0:PSET(39,V)CHR$(127),6,6:PSET(39,
B)CHR$(127),6,6:PSET(BA,21)CHR$(127),6,6:PSET(BA+1
,21)CHR$(127),6,6:GOTO 600
330 SC=SC+(17-V)*(1-(DB<0)):LOCATE 10,0,0:COLOR 0,
6:PRINT SC:FLAG=0:PSET(I,V)CHR$(127),6,6
340 FOR J=0 TO 7
```

```
350 LINE(BA*8,21*8+J)-(BA*8+16,21*8+J),6
360 NEXT J
370 GOTO 600
400 PSET(39,V)GR$(0),0
410 FOR I=38 TO 0 STEP-1
420 PSET(I+1,V)CHR$(127),6,6:PSET(I,V)GR$(0),0
430 IF FLAG=1 THEN GOTO 450
440 IF INKEY$="F" THEN FLAG=1:PSET(I,V+1)GR$(4),1:
B=V+1:GOTO 470 ELSE GOTO 470
450 PSET(I+1,B)CHR$(127),6,6:B=B+1:PSET(I,B)GR$(4)
,1:IF B<21 THEN GOTO 470
460 IF I=BA OR I=BA+1 THEN GOTO 490 ELSE FLAG=0:PS
ET(I,21)CHR$(127),6,6
470 IF (I MOD3=0)THEN PSET(BA-(DB<0),21)CHR$(127),
6,6:BA=BA+DB:PSET(BA,21)GR$(2),0:PSET(BA+1,21)GR$(
3),0
480 NEXT I:FLAG=0:PSET(0,V)CHR$(127),6,6:PSET(0,B)
CHR$(127),6,6:PSET(BA,21)CHR$(127),6,6:PSET(BA+1,2
1)CHR$(127),6,6:GOTO 600
490 SC=SC+(17-V)*(1-(DB>0)):LOCATE 10,0,0:COLOR 0,
6:PRINT SC:FLAG=0:PSET(I,V)CHR$(127),6,6
500 FOR J=0 TO 7
510 LINE(BA*8,21*8+J)-(BA*8+16,21*8+J),6
520 NEXT J
600 LINE(19,2)-(19,18)CHR$(127),6,6:NEXT K
700 COLOR 0,4:LOCATE 0,24:INPUT"VOULEZ VOUS REJOUE
R(O/N)";REP$
710 IF LEFT$(REP$,1)="N" THEN SCREEN 4,6,6:CLS:END
 ELSE GOTO 100
1000 SCREEN 6,6,0:CLS
1010 BOXF(0,22)-(39,24)CHR$(127),4
1020 RETURN
2000 SCREEN ,0,0:CLS
2010 BOXF(0,15)-(39,24)CHR$(127),4
2020 COLOR 1:ATTRB 1,1
2030 LOCATE 10,14,0:PRINT GR$(2);GR$(3)
2040 COLOR 7:LOCATE 6,4:PRINT GR$(1)
2050 COLOR 3:LOCATE 15,3:PRINT"BOMBARDIER"
2060 ATTRB 0,0:FOR I=0 TO 3
2070 PSET(5-I,6+I)GR$(5),2:NEXT I
2090 COLOR 2,4:LOCATE 0,23
2100 PRINT "APPUYER SUR ENTREE"
2110 IF INKEY$<>CHR$(13) THEN GOTO 2110
2120 RETURN
```

4.3 D.C.A.

Vous êtes le seul artilleur du secteur et 50 avions ennemis vont tenter de passer au-dessus de votre batterie de fort calibre. Ceux qui réussiront à passer pourront infliger des pertes supplémentaires aux vôtres dont vous êtes localement l'unique défenseur. Les avions paraissent à des altitudes différentes ce qui rend la tâche plus rude car un obus de D.C.A. doit toujoursêtre réglé en fonction du niveau de vol de l'avion.

Fonctionnement du jeu

La commande RUN fait apparaître la page de présentation. L'espace de combat est affiché lors d'une action sur la touche [ENTREE]. Pour ouvrir le feu, il faut utiliser la touche [F] du clavier ; si vous parvenez à toucher un avion, celui-ci tombera ; sinon, il continuera son vol jusqu'à sa sortie de l'écran. Le score sera d'autant plus augmenté que l'avion touché sera plus haut.

Structure du programme

Les lignes 10 à 60 définissent les différents caractéres graphiques utilisés (batterie, obus, avions dans les deux sens de déplacement et avion en piqué).

Les lignes 220 à 370 assurent le déplacement de la gauche vers la droite de l'avion ainsi que le tir de D.C.A. (ligne 260), le déplacement de l'obus (lignes 270 et 280).

Les lignes 320 à 350 assurent la chute de l'avion lorsque celui-ci est touché.

Les lignes 400 à 560 jouent le même rôle que le bloc précédent mais pour un déplacement de droite à gauche de l'appareil.

Préalablement, la ligne 210 a provoqué de façon aléatoire un branchement vers la ligne 220 ou la ligne 400.

La ligne 600 efface éventuellement un obus qui serait resté en l'air alors que l'avion sort de l'écran et renvoie au début de la boucle qui compte les 50 passages.

Les lignes 700 et 710 permettent de rejouer ou de sortir du programme en fin de partie.

Les lignes 1000 à 1040 dessinent l'espace de combat.

Les lignes 2000 à 2130 assurent l'affichage de la page de présentation.

```
10 CLEAR,,5
20 DEFGR$(0)=0,3,131,239,252,160,48,48
30 DEFGR$(1)=24,24,24,24,60,126,255,66
40 DEFGR$(2)=24,24,60,60,60,60,60,126
50 DEFGR$(3)=0,192,193,255,63,5,12,12
60 DEFGR$(4)=60,24,24,24,255,255,24,24
70 GOSUB 2000
100 SC=0
110 GOSUB 1000
120 COLOR 4,6:LOCATE 0,0,0:PRINT "APPUYER SUR ENTR
EE POUR COMMENCER"
130 A=RND
140 IF INKEY$<>CHR$(13)THEN GOTO 130
150 LOCATE 0,0,0:PRINT SPC(39)
200 FOR K=1 TO 50
210 V=2+INT(RND*14):IF RND>0.5 THEN GOTO 400
220 PSET(0,V)GR$(3),0
230 FOR I=1 TO 39
240 PSET(I-1,V)CHR$(127),6,6:PSET(I,V)GR$(3),0
250 IF FLAG=1 THEN GOTO 270
```

```
260 IF INKEY$="F" THEN FLAG=1 :PSET(19,18)GR$(2),1
:M=18:ELSE GOTO 290
270 PSET(19,M)CHR$(127),6,6:M=M-1:PSET(19,M)GR$(2)
,1:IF M>V THEN GOTO 290
280 IF I=19 THEN GOTO 300 ELSE FLAG=0:PSET(19,V)CH
R$(127),6,6
290 NEXT I:PSET(39,V)CHR$(127),6,6:GOTO 600
300 SC=SC+17-V:LOCATE 11,23,0:COLOR 4,5:PRINT SC
310 FLAG=0:PSET(19,V)CHR$(127),6,6
320 FOR I=V TO 19
330 PSET(20+I-V,I)GR$(4),0:PSET(19+I-V,I-1)CHR$(12
7),6,6
340 FOR J=1 TO 5:NEXT J
350 NEXT I
360 PSET(19+I-V,19)CHR$(127),6,6
370 GOTO 600
400 PSET(39,V)GR$(0),0
410 FOR I=38 TO 0 STEP -1
420 PSET(I+1,V)CHR$(127),6,6:PSET(I,V)GR$(0),0
430 IF FLAG=1 THEN GOTO 450
440 IF INKEY$="F" THEN FLAG=1 :PSET(19,18)GR$(2),1
:M=18:ELSE GOTO 470
450 PSET(19,M)CHR$(127),6,6:M=M-1:PSET(19,M)GR$(2)
,1:IF M>V THEN GOTO 470
460 IF I=19 THEN GOTO 500:ELSE FLAG=0:PSET(19,V)CH
R$(127),6,6
470 NEXT I:PSET(0,V)CHR$(127),6,6:GOTO 600
500 SC=SC+17-V:LOCATE 11,23,0:COLOR 4,5:PRINT SC
510 FLAG=0:PSET(19,V)CHR$(127),6,6:PSET(1,V)CHR$(1
27),6,6
520 FOR I=V TO 19
530 PSET(18-I+V,I)GR$(4),0:PSET(19-I+V,I-1)CHR$(12
7),6,6
540 FOR J=1 TO 5:NEXT J
550 NEXT I
560 PSET(19-I+V,19)CHR$(127),6,6
600 LINE(19,2)-(19,18)CHR$(127),6,6:NEXT K
700 COLOR 4,5:LOCATE 0,24:INPUT"VOULEZ VOUS REJOUE
R(O/N)";REP$
710 IF LEFT$(REP$,1)="N" THEN SCREEN 4,6,6:CLS:END
 ELSE GOTO 100
1000 SCREEN 6,6,6:CLS
1010 BOXF(0,20)-(39,24)CHR$(127),5
1020 PSET(19,19)GR$(1),0
1030 COLOR 4,5:LOCATE 0,23,0:PRINT"SCORE":LOCATE 1
1,23:PRINT SC
1040 RETURN
2000 'PRESENTATION
2010 SCREEN 2,0,0:CLS
2020 ATTRB 1,1:LOCATE 20,10,0:PRINT"D.C.A."
```

```
2030 LOCATE 30,15:COLOR 4:PRINT GR$(0)
2040 LINE(56,160)-(152,80),1
2050 LINE(192,160)-(200,84),1
2060 LINE(300,160)-(250,80),1
2070 LOCATE 3,5:PRINT GR$(3)
2080 LOCATE 7,7:PRINT GR$(3)
2090 LOCATE 3,12:COLOR 1:PRINT GR$(2)
2100 LOCATE 3,16:COLOR 1:PRINT GR$(2)
2110 ATTRB 0,0:COLOR 3:LOCATE 0,23:PRINT"APPUYER S
UR ENTREE"
2120 IF INKEY$<>CHR$(13)THEN GOTO 2120
2130 RETURN
```

4.4 LETTRIVORE (VERSION CLAVIER)

Bien que LETTRIVORE soit un jeu particulièrement amusant à jouer avec un joystick, nous avons pensé qu'il était tout de même intéressant de donner une version "clavier" de ce jeu. Il suffit simplement pour cela de modifier le sous-programme de jeu.

Vous ne trouverez donc ci-dessous que le sous-programme de jeu, le reste du programme ainsi que les explications détaillées concernant sa structure et les règles du jeu étant dans le chapitre des jeux avec joystick.

Pour déplacer la chenille, pressez les touches [→], [←], [↑] et [↓].

Structure du sous-programme de jeu (version clavier) : lignes 1 000 à 1 120.

REMARQUE : les lignes 1 200 à 1 550 font partie de ce sous-programme mais elles restent identiques à celles de la version joystick.

Ligne 1000, place la chenille au centre de l'écran - la lettre A doit être la première lettre mangée - la chenille se déplace automatiquement de gauche à droite au début du jeu.

Ligne 1010, lecture du clavier - la chenille garde la même direction si aucune nouvelle touche n'a été pressée

Ligne 1020, mise en mémoire de la position actuelle de la chenille - le chronomètre T est incrémenté.

Lignes 1030 et 1040, calcul des nouvelles coordonnées de la chenille.

Lignes 1050 à 1120, cf lignes 1110 à 1180 de la version joystick.

```
1000 PSET(15,12)"*":P=65:B$=CHR$(9)
1010 A$=INKEY$:IF A$="" THEN A$=B$
1020 A=ASC(A$):AH=H:AV=V:T=T+1
1030 H=H+(A=8)-(A=9)
1040 V=V+(A=11)-(A=10)
1050 IF H=39 OR H=0 THEN H=ABS(H-38)
1060 IF V=24 OR V=0 THEN V=ABS(V-23)
1070 C=SCREEN(H,V):IF C=32 OR C=42 THEN 1110
1080 IF C<>P THEN 1200
1090 S=S+1:PLAY P$
1100 P=P+1:IF P=91 THEN 1500
1110 PSET(AH,AV)" ":PSET(H,V)"*":LOCATE7,0,0:PRINT
 T
1120 FORI=1 TO N*10:NEXT I:B$=A$:GOTO 1010
1200 FORI=1 TO 10:PLAY P$:NEXTI:PSET(H,V)" "
1210 LOCATE0,0,0:PRINT"VOUS AVEZ PERDU..."
1220 LOCATE0,5,0:PRINT"VOUS AVEZ MANGE";S;"LETTRES
"
1230 GOTO 1550
1500 FORI=1 TO 10:PLAY P$:NEXTI:PSET(H,V)" "
1510 LOCATE0,0,0:PRINT"VOUS AVEZ GAGNE EN";T;"'SEC
ONDES'"
1520 LOCATE0,5,0
1530 IF T<RECORD(N) THEN PRINT"LE RECORD AU NIVEAU
";N;"EST MAINTENANT: ";T:RECORD(N)=T:RETURN
1540 PRINT"LE RECORD AU NIVEAU";N;"EST TOUJOURS: "
:RECORD(N)
1550 RETURN
2000 CLS:INPUT"NIVEAU(0-9)";N:N=INT(ABS(N)):IF N>9
 THEN 2000
```

4.5 MISSILES

Une vague d'envahisseurs extra-terrestres, se dirigeant à grande vitesse vers la terre, vient d'être repérée dans l'espace. Devant ce péril effroyable qui menace le monde entier, le gouvernement a donné l'ordre à ses meilleurs pilotes de fusées de décoller aussitôt pour tenter de repousser l'ennemi.

Vous faites partie de ces pilotes d'élite et vous voilà aux commandes de votre astronef TO7, équipé -pour accomplir cette mission dangereuse-de trente missiles.

Dernier cri de la technologie moderne, votre TO7 est pourvu d'un pilotage automatique, ce qui vous permet de combattre dans les meilleures conditions possibles et de vous concentrer uniquement sur le tir de vos missiles. Dès que vous serez à la verticale d'un adversaire, il vous suffira de presser la touche [F] pour lancer un missile dans sa direction.

Chaque envahisseur touché rapporte cinq points. Si vous arrivez à repousser la première vague de vingt envahisseurs (ce qui est une belle performance avec seulement trente missiles), un bonus de cinquante points vous est octroyé. Votre vaisseau est alors réarmé avec trente missiles et vous repartez aussitôt pour affronter une nouvelle vague d'envahisseurs.

Le jeu s'arrête dès que vos munitions sont épuisées alors qu'il reste des envahisseurs. Votre vaisseau, qui était invulnérable jusqu'à cet instant, est en effet à la merci de l'ennemi.

Pour rejouer, pressez soit la touche [O] soit simplement la touche [ENTREE]. La touche [N] arrête définitivement le jeu.

Structure du programme

Programme principal et fin de partie : lignes 1 à 110 et lignes 2000 à 2030.

Ligne 1, réservation de place mémoire pour trois caractères définis - appel du sous-programme de présentation.

Ligne 5, effacement de l'écran - compteur du nombre d'envahisseurs détruits (N) et score (S) à zéro - nombre de missiles (B) à 30. Appel du sous-programme d'affichage.

Lignes 10 à 50, déplacement de la fusée à droite - lecture du clavier et déclenchement du tir par appel au sous-programme de tir si la touche est appuyée (ligne 30).

Lignes 60 à 100, cf lignes 10 à 50 mais déplacement de droite à gauche.

Ligne 110, retour en 10 pour un nouveau passage.

Lignes 2000 à 2030, fin de partie classique.

Sous-programme de tir : lignes 1000 à 1100

Ligne 1000, choix de l'attaque, tempo, durée et octave des notes - décrément du compteur de missiles.

Ligne 1010, affichage du nombre de missiles restants.

Lignes 1020 à 1060, tir du missile avec bruit approprié (ligne 1040).

Ligne 1070, teste si le missile a touché ou non un envahisseur - si ce n'est pas le cas, appel du sous-programme de test puis retour au programme principal.

Ligne 1080, enlève l'envahisseur détruit.

Ligne 1090, incrémentation du score et affichage avec sonorisation.

Ligne 1700, incrémentation du compteur d'envahisseurs détruits - appel du sous-programme de test.

Ligne 1110, retour au programme principal.

Sous-programme de test : lignes 1500 à 1580

Ligne 1500, renvoi à la fin si tous les missiles ont été tirés.

Ligne 150, retour au programme principal s'il reste encore des envahisseurs à détruire.

Ligne 1520, affichage d'un message signalant le gain du bonus.

Ligne 1530, incrémentation du score et bip sonore.

Ligne 1540, effacement du message de bonus.

Ligne 1560, nombre d'envahisseurs détruits remis à zéro et nombre de missiles remis à trente

Ligne 1570, appel du sous-programme d'affichage.

Ligne 1580, retour au programme principal.

Sous-programme d'affichage : lignes 2500 à 2560

Lignes 2500 à 2520, affichage de vingt envahisseurs en haut de l'écran

Lignes 2530 à 2550, affichage du score et du nombre de missiles.

Lignes 2560, retour au programme principal.

Sous-programme de présentation : lignes 3000 à 3260

Ligne 3000, effacement de l'écran - déclaration d'un tableau de chaînes de caractères qui contiendra les notes de la gamme chromatique. Ce tableau sera lu, élément par élément, lors de la montée d'un missile, ce qui permettra un bruitage approprié - le bonus est de cinquante points.

Ligne 3010, couleur de l'écran : noir - couleur des caractères : vert.

Ligne 3020, la variable C$ contient les notes de la gamme chromatique. Cette variable sera utilisée pour le bruitage lors de l'incrémentation du score.

Ligne 3030, la variable B$ contient les paramètres qui règlent l'attaque, le tempo et la durée des notes.

Lignes 3040 à 3070, lecture des données en data, ce sont les notes de la gamme chromatique.

Lignes 3080 à 3100, définition de trois caractères : envahisseur, fusée, missile.

Ligne 3110, caractère en double hauteur et double largeur - couleur des caractères : bleu.

Ligne 3120, affichage au centre de l'écran du nom du jeu.

Lignes 3130 à 3160, affichage d'envahisseurs rouges dans la partie supérieure de l'écran.

Ligne 3170, affichage d'une fusée verte dans le coin inférieur gauche de l'écran.

Lignes 3180 à 3220, tir d'un missile.

Ligne 3230, retour à des caractères de taille normale et de couleur verte - bruitage signifiant qu'un des envahisseurs vient d'être touché (admirez au passage la précision du tir ... soigneusement calculé !!).

Ligne 3240, attend la pression de la touche [ENTREE] pour laisser le jeu commencer.

Ligne 3250, effacement de l'écran.

Ligne 3260, retour au programme principal.

```
1 CLEAR,,3:GOSUB3000
5 CLS:S=0:N=0:B=30:GOSUB2500
10 FORI=0TO39
20 LOCATEI,20:PRINTGR$(1)
30 A$=INKEY$:IF A$="F"THENGOSUB1000
40 LOCATEI,20:PRINT" "
50 NEXTI
60 FORI=39TO0STEP-1
70 LOCATEI,20:PRINTGR$(1)
80 A$=INKEY$:IF A$="F"THENGOSUB1000
90 LOCATEI,20:PRINT" "
100 NEXTI
110 GOTO10
1000 PLAY"A011L404":B=B-1
1010 COLOR6,0:LOCATE35,22,0:PRINTB:COLOR2,0
1020 FORJ=19 TO1STEP-1
1030 LOCATEI,J:PRINT GR$(2)
1040 PLAYA$(J)
1050 LOCATEI,J,0:PRINT" "
```

```
1060 NEXTJ
1070 IFPOINT(4+8*I,4)<>1THEN GOSUB1500:RETURN
1080 LOCATEI,0,0:PRINT" ":COLOR6,0
1090 FORJ=1TO5:S=S+1:LOCATE7,22,0:PRINTS:PLAYB$+C$
:NEXTJ
1100 N=N+1:COLOR2,0:GOSUB1500
1110 RETURN
1500 IF B=0 THEN 2000
1510 IF N<>20 THEN RETURN
1520 LOCATE15,22,0:COLOR1,0:PRINT"BONUS: ";BONUS
1530 S=S+BONUS:FORJ=1 TO10:BEEP:FORK=1TO50:NEXTK:N
EXTJ
1540 LOCATE15,22,0:PRINT"
1560 N=0:B=30
1570 GOSUB2500
1580 RETURN
2000 LOCATE0,10:INPUT"VOULEZ-VOUS REJOUER ";REP$
2010 IF LEFT$(REP$,1)="O" OR REP$="" THEN 5
2020 IF LEFT$(REP$,1)="N" THEN CLS:END
2030 GOTO 2000
2500 FORI=0 TO 38 STEP 2
2510 LOCATEI,0:COLOR1,0:PRINT GR$(0)
2520 NEXTI
2530 COLOR5,0:LOCATE0,22:PRINT"SCORE: ":LOCATE25,2
2:PRINT"MISSILES: "
2540 COLOR6,0:LOCATE7,22:PRINT S
2550 LOCATE35,22:PRINT B:COLOR2,0
2560 RETURN
3000 CLS:DIMA$(26):BONUS=50
3010 SCREEN2,0,0
3020 C$="DODO#RERE#MIFAFA#SOSO#LALA#SI"
3030 B$="A0T1L4"
3040 FORI=1TO21
3050 READ A$(I)
3060 NEXTI
3070 DATA "SO","FA#","FA","MI","RE#","RE","DO#","D
O","O5","SI","LA#","LA","SO#","SO","FA#","FA","MI"
,"RE#","RE","DO#","DO"
3080 DEFGR$(0)=60,126,219,255,255,153,153,153
3090 DEFGR$(1)=24,24,24,60,126,255,189,153
3100 DEFGR$(2)=24,24,60,102,189,153,24,60
3110 ATTRB1,1:COLOR4,0
3120 LOCATE10,12:PRINT"MISSILES"
3130 LOCATE0,2:COLOR1,0
3140 FORI=0TO9
3150 PRINT GR$(0);" ";
3160 NEXTI
3170 COLOR2,0:LOCATE0,22:PRINT GR$(1)
3180 FORI=20 TO 2 STEP -2
3190 LOCATE0,I,0:COLOR3,0:PRINT GR$(2)
```

```
3200 PLAY"L2DO"
3210 LOCATE0,I,0:PRINT" "
3220 NEXTI
3230 ATTRB0,0:COLOR2,0:FORJ=1TO10:PLAYB$+C$:NEXTJ
3240 LOCATE0,24:INPUT"PRESSEZ ENTREE POUR COMMENCE
R",Q$
3250 RETURN
```

4.6 PICKMAN (VERSION CLAVIER)

Le programme PICKMAN ainsi que les commentaires associés se trouvent au chapitre 5. Nous proposons dans ce paragraphe une modification du programme (utilisé normalement avec des manettes de jeu) permettant de l'utiliser dans de bonnes conditions avec le clavier.

Les lignes 210 à 320 du chapitre 5 doivent être remplacées par les 8 lignes ci-après (210 à 270).

Les touches suivantes sont utilisées pour commander le déplacement :

[↑] commande un déplacement vers le haut

[↓] commande un déplacement vers le bas

[←] commande un déplacement vers la gauche

[→] commande un déplacement vers la droite

Fonctionnement du bloc

Ce bloc de déplacement du PICKMAN remplace celui du chapitre 5 exactement de la même façon qu'une carte enfichable dans un système électronique.

La ligne 210 transfère dans un premier temps la commande issue du clavier dans la variable chaîne IM$. Ceci a une double raison d'être, tout d'abord le fait d'utiliser INKEY$ plusieurs fois donne un résultat erroné (puisque INKEY$ représente une sorte de pile qui contient en son sommet la dernière touche enfoncée et qui se dépile à chaque lecture, c'est-à-dire à chaque INKEY$) et ensuite, l'instruction INKEY$ est très lente, il est plus rapide d'utiliser une autre variable chaîne).

Dans un deuxième temps, si le clavier n'a pas été utilisé, on passe directement au déplacement des monstres sinon, le caractère graphique représentant la bonne direction du PICKMAN est sélectionné.

La ligne 220 interprète la commande issue du clavier (IM$) et recherche la couleur de graphisme de la case où doit aller le PICKMAN.

Si cette couleur est le mauve (oeuf) le score est incrémenté, par contre s'il s'agit du bleu ou du rouge aucun déplacement n'est autorisé (230 à 250).

La ligne 260 vérifie que tous les oeufs n'ont pas été dévorés.

La ligne 270 a la même raison d'être que la ligne 320 de la version "manette de jeu". Les valeurs (0.5 et 0.4) sont différentes car il faut rétablir les proportions temporelles en fonction des temps de réponse clavier et manette de jeu.

REMARQUE : les lignes 200 et 500 du listing ci-dessous vous serviront de point de repère pour la modification proposée.

```
200 'DEPLACEMENT DU PICKMAN
210 IM$=INKEY$:IF IM$="" THEN GOTO 510 ELSE C=ASC(
IM$):PAC$=GR$(C-6)
220 V=PH:PH=PH+(C=8)-(C=9):W=PV:PV=PV+(C=11)-(C=10
):YL=POINT(PH*8+3,PV*8+3)
230 OEUF=OEUF+(YL=5):IF YL=4 OR YL=1 THEN PH=V:PV=
W
240 PSET(V,W)CHR$(127),0:PSET(PH,PV)PAC$,2
250 SCORE=SCORE-17*(YL=5):LOCATE 0,13,0:PRINT SCOR
E
260 IF OEUF=0THEN GOTO 1500
270 IF RND>0.4 THEN GOTO 210
500 'DEPLACEMENT DES MONSTRES
```

4.7 STOCK-CAR

Vous voilà "manager" d'une écurie de cascade très spéciale. Vos pilotes au volant de leurs bolides mettent à l'épreuve leurs nerfs et leur habileté de conducteur face à une voiture-robot intraitable.

Les foules ont pris goût au spectacle de ce jeu dangereux et les billets d'accés aux gradins s'arrachent à prix d'or. Cependant, pour maintenir l'intérêt que suscite auprès du public une telle activité, vos pilotes doivent aller toujours plus loin. Désormais, ils ne pourront plus voir par eux-mêmes la voiture-robot qui cherche la collision de front à vitesse maximale.

Dans le but d'éviter une prestation fort brève à vos voitures, vous avez installé à bord un récepteur radio, par lequel vous informez les cascadeurs de la direction à prendre, puisqu'à l'extérieur, on a une vue d'ensemble de la scène.

Règle du jeu

La mesure des performances se fait d'une manière très simple : des croix lumineuses sont disposées sur

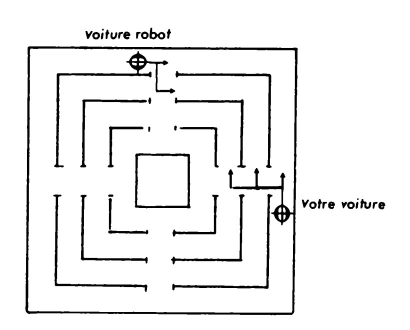
la route à intervalles réguliers et celles-ci s'éteignent lors du passage de la voiture ; chaque extinction augmente le score d'un point. Lorsque l'automobile a emprunté toutes les voies possibles, et que la dernière croix vient d'être éteinte, la voiture retourne au garage en révision, le pilote est changé et un nouveau véhicule mené par un autre pilote prend la relève tandis que toutes les croix lumineuses se rallument.

En cas d'accident, les croix gardent leur état et un nouveau bolide succède au précédent.

Vous disposez de trois voitures dont les scores sont cumulés.

La scène de cette ronde infernale est désormais classique : quatre pistes concentriques forment des carrés, il n'est possible de changer de piste qu'au milieu de chaque ligne droite. La voiture robot ne peut changer que d'une piste par "carrefour" alors que la voiture dirigée par un humain a droit à deux pistes.

Bien entendu, il est impossible de se croiser sur une même piste et les murs sont suffisamment hauts pour empêcher le pilote de voir l'ensemble. Les voitures ne peuvent ni faire demi-tour, ni revenir en arrière et roulent en sens inverse.



Exemple des possibilités de déplacement

Fonctionnement du programme

Après avoir lancé le programme à l'aide de la commande RUN, la page de présentation apparaît durant 10 secondes. Ce temps est utilisé au chargement et au calcul d'un tableau 25 x 25. Puis le jeu commence. Une première voiture quitte son stationnement pour rejoindre le pilote de départ, un compte à rebours partant de 5 et battant la seconde s'entame (l'affichage a lieu dans le carré central de l'écran) ; au terme de ce décomptage, les voitures démarrent.

La voiture verte est dirigée par les touches [←] et [→] et la voiture rouge représente la voiture-robot ; les deux touches s'utilisent de la façon suivante :

La touche [←] commande un déplacement vers l'intérieur du carré.

La touche [→] commande un déplacement vers l'extérieur.

Lors d'une collision, la voiture rouge reprend la position de départ tandis qu'une nouvelle auto verte sort du parking. Le parking possède trois places.

Si toutes les croix sont effacées, un nouveau tableau apparaît et la voiture repart du parking.

En fin de partie, un message s'affiche à l'intérieur du carré central vous invitant à rejouer (presser [ENTREE]) ou à sortir du programme (presser [N] puis [ENTREE]).

Structure du programme

Bloc principal : 0 à 170

Ce bloc comprend l'initialisation des tableaux, des caractères graphiques, des scores, des positions des voitures et de toutes les autres variables nécessaires. La plupart des fonctions sont remplies par les sous-programmes appelés, décrits ci-dessous.

Sous-programme de présentation du jeu : 6000 à 6240

Son rôle est très classique, il se contente d'afficher la page de présentation qui représente un tronçon de route avec pointillés jaunes.

Sous-programme de création du tableau de travail

Le bloc de déplacement des voitures utilise un tableau 25 x 25. Dans le but de gagner du temps lors de l'écriture du programme mais aussi de la place en mémoire (ce programme est à la limite de la saturation de l'espace mémoire de la version de base du TO7), la symétrie du tableau ROU (pour Route) a été utilisée. Ainsi, seul le quart du tableau figure sous forme de "DATA" dans le listing.

Les lignes 10000 à 10100 reconstituent le tableau ROU à partir des données se trouvant aux lignes 10200 à 10310.

Sous-programme de tracé du terrain de jeu

Les rectangles concentriques (bleus) sont tracés par les lignes 1070 à 1130 puis les carrefours sont "creusés" aux lignes 1200 à 1215 et les croix jaunes sont alors réparties sur les quatre pistes (1220 à 1240). Auparavant, le garage a été dessiné (1010 à 1060). Le sous-programme s'achève après avoir déposé les trois voitures vertes à leurs places respectives sur le parking.

Sous-programme d'initialisation : 2000 à 2100

Les variables utilisées durant le jeu prennent leurs valeurs initiales dans cette partie. On y trouve des données permettant de connaître la position de départ des automobiles, les déplacements horizontaux et verticaux ainsi que l'affichage du score et du record.

Sous-programme de déplacement vers le plot de départ 3000 à 3390

Le mouvement des voitures entre le garage et la position de départ ainsi que l'ouverture et la fermeture de la porte sont pris en charge par cette partie du programme. Le compte à rebours a été inséré à la fin (3300 à 3340).

Bloc de déplacement des voitures : 200 à 910

Bien que relativement court, c'est ce bloc qui assure le déroulement du jeu ; il est composé de deux sous-blocs, l'un chargé du mouvement de la voiture joueur et de son déplacement en fonction des touches enfoncées et l'autre chargé du déplacement de la voiture robot ainsi que de la recherche de la collision.

En fonction de la position de chacune des voitures, un mouvement particulier doit être envisagé, par exemple le virage automatique dans les angles de la piste, le sens de déplacement (vers le haut, le bas, la droite ou la gauche), l'autorisation pour les voitures de changer de piste (aux carrefours) et bien d'autres détails. Malgré l'ensemble des tests qu'il y a à effectuer lors de chaque déplacement élémentaire, le jeu doit rester rapide. Pour ces différentes raisons, la solution adoptée a été celle du tableau ; ainsi , à chacune des cases de l'écran correspond une

case (dans le tableau ROU) qui contient un chiffre compris entre 0 et 9. A l'aide de l'instruction ON GOTO, il est alors simple d'aiguiller rapidement le programme vers une des 10 options que comprend un jeu de ce type. Le lecteur intéressé par les détails du programme remarquera aux lignes 10200 à 10310 la disposition de ces chiffres dans le tableau, il est facile de deviner l'utilité de chacun d'entre eux.

Deux tableaux auraient pu être nécessaires, mais la symétrie du problème associée aux possibilités d'options fournies par ON GOTO permet de n'en utiliser qu'un.

Les lignes 200 à 590 permettent le déplacement de la voiture du joueur ainsi que l'incrémentation du score.

Les lignes 600 à 910 se chargent quant à elles, du mouvement de la voiture rouge (voiture robot).

<u>Bloc "Partie supplémentaire"</u> : 5000 à 5050

Dans ce bloc, les croix sont réaffichées lorsque le joueur est parvenu à toutes les effacer.

<u>Bloc de fin de jeu</u> : 4000 à 4050

Il constitue la fin classique des jeux de cet ouvrage. Une question est posée à l'utilisateur sur sa volonté de continuer, et suivant la réponse, la tâche est abandonnée ou reprise. En cas de recommencement d'une partie, le nouveau record prend la place du précédent s'il y a lieu.

Remarque concernant la recopie du jeu à l'aide du clavier : ce jeu constitue le programme le plus long de ce recueil, aussi nous conseillons aux débutants de rentrer minutieusement chaque ligne en la vérifiant plusieurs fois car il sera bien difficile (mais très instructif) de détecter une erreur parmi quelques 160 lignes.

Soyez donc très patient, et bonne chance face à cette terrible voiture-robot.

```
0 'STOCK CAR
1 CLEAR,,5:DIM ROU%(25,25):DEFINT A-Z
10 CONSOLE 0,24:SCREEN 2,0,0:CLS
20 DEFGR$(0)=0,231,126,255,255,126,231,0
30 DEFGR$(1)=90,126,126,60,60,126,126,90
40 DEFGR$(2)=24,24,24,24,24,24,24,24
50 DEFGR$(3)=1,3,6,12,24,48,0,0
60 DEFGR$(4)=24,24,24,24,24,24,255,255
70 GOSUB 6000
90 GOSUB 10000
100 MSC=200
110 SC=0
150 GOSUB 1000
160 GOSUB 2000
170 GOSUB 3000
200 'DEPLACEMENT DES VOITURES
250 V=TH:W=TV:ON ROU(TH,TV) GOTO 560,510,550,300,3
50,400,350,400,300
260 IM$=INKEY$:S=(IM$=CHR$(9))-(IM$=CHR$(8)):S=S+(
IM$=""):TH=TH+S*AV:TV=TV-S*AH:GOTO 560
300 IM$=INKEY$:S=(IM$=CHR$(9))-(IM$=CHR$(8)):TH=TH
+S*AV:TV=TV-S*AH:GOTO 560
350 B=(INKEY$=CHR$(8)):TH=TH-B*AV:TV=TV+B*AH:GOTO
560
400 B=(INKEY$=CHR$(9)):TH=TH+B*AV:TV=TV-B*AH:GOTO
560
510 TH=TH+AH:TV=TV+AV:SC=SC-(SCREEN(TH+8,TV)=43)
520 PSET(V+8,W)CHR$(127),0:PSET(TH+8,TV)V$,2:V=TH:
W=TV:IF(TH=UH)AND(TV=UV)THEN GOTO 2900 ELSE GOTO 5
60
550 A=AV:AV=(AH=1)-(AH=-1):AH=(A=-1)-(A=1):V$=GR$(
-(AH=0)):R$=INKEY$
560 TH=TH+AH:TV=TV+AV:SC=SC-(SCREEN(TH+8,TV)=43)
```

```
570 PSET(V+8,W)CHR$(127),0:PSET(TH+8,TV)V$,2:LOCAT
E 0,4:PRINT SC
580 IF TH=UH THEN IF TV=UV THEN GOTO 2900
590 IF SC MOD 255=0 THEN GOTO 5000
600 V=UH:W=UV:ON ROU(24-UH,UV)GOTO 900,900,850,650
,700,750,900,900,900
610 L=ABS(TH-12):M=ABS(TV-12):N=ABS(UH-12):P=ABS(U
V-12):T=(L>=M)*L+(M>L)*M-(N>=P)*N-(P>N)*P:T=SGN(T)
+(T=0):UH=UH-EV*T:UV=UV+EH*T:GOTO 900
650 L=ABS(TH-12):M=ABS(TV-12):N=ABS(UH-12):P=ABS(U
V-12):T=(L>=M)*L+(M>L)*M-(N>=P)*N-(P>N)*P:T=SGN(T)
:UH=UH-EV*T:UV=UV+EH*T:GOTO 900
700 L=ABS(TH-12):M=ABS(TV-12):N=ABS(UH-12):P=ABS(U
V-12):T=-((L>=M)*L+(M>L)*M-(N>=P)*N-(P>N)*P>0):UH=
UH-EV*T:UV=UV+EH*T:GOTO 900
750 L=ABS(TH-12):M=ABS(TV-12):N=ABS(UH-12):P=ABS(U
V-12):T=((L>=M)*L+(M>L)*M-(N>=P)*N-(P>N)*P<0):UH=U
H-EV*T:UV=UV+EH*T:GOTO 900
850 E=EV:EV=(EH=-1)-(EH=1):EH=(E=1)-(E=-1):E$=GR$(
-(EH=0)):GOTO 900
900 UH=UH+EH:UV=UV+EV:PSET(V+8,W)CHR$(Q),-(Q=43)*3
:Q=SCREEN(UH+8,UV):PSET(UH+8,UV)E$,1
910 IF(TH=UH)AND(TV=UV) THEN GOTO 2900 ELSE GOTO 2
50
1000 'TRACE DU JEU
1010 SCREEN2,0,0:CLS:LINE(4,160)-(52,160),4
1020 BOX(0,152)-(67,196),4
1030 FOR I=0 TO 3
1040 LINE(4+I*16,160)-(4+I*16,176),4
1060 NEXT I
1070 FOR I=0 TO 4
1080 FOR J=0 TO 1
1090 BOX(67+I*16+J,3+I*16+J)-(260-I*16-J,196-I*16-
J),4
1120 NEXT J
1130 NEXT I
1200 BOXF(9,11)-(15,13)CHR$(127),0
1205 BOXF(25,11)-(31,13)CHR$(127),0
1210 BOXF(19,1)-(21,7)CHR$(127),0
1215 BOXF(19,23)-(21,17)CHR$(127),0
1220 FOR I=0 TO 3
1230 BOX(9+I*2,1+I*2)-(31-I*2,23-I*2)"+",3
1240 NEXT I
1250 FOR I=0 TO 2
1260 PSET (1+I*2,21)GR$(1),2
1270 NEXT I
1280 RETURN
2000 'INITIALISATION
2010 TH=2:TV=23:AV=0:AH=1:Q1=43:V$=GR$(0)
2030 UH=23:UV=16:EH=0:EV=1:Q=43:E$=GR$(1)
2040 Q=SCREEN(UH+8,UV):PSET(UH+8,UV)E$,1
```

```
2050 LOCATE 0,3,0:COLOR 2:PRINT"SCORE"
2060 LOCATE 2,4:PRINT"0"
2070 LOCATE 0,8:PRINT MSC
2080 LOCATE 0,7:PRINT"RECORD"
2090 NBVR=3
2100 RETURN
2900 'COLLISION
2910 PSET(TH+8,TV)CHR$(127),0
2920 GOTO 170
3000 'SORTIE DU GARAGE
3010 IF NBVR=0 THEN GOTO 4000
3015 PSET(TH+8,TV)CHR$(Q1),-(Q1=43)*3
3016 PSET(UH+8,UV)CHR$(Q),-(Q=43)*3
3017 Q=SCREEN(31,16):PSET(31,16)GR$(1),1
3020 FOR I=0 TO 1
3030 PSET(1+(3-NBVR)*2,21+I)CHR$(127),0
3040 PSET(1+(3-NBVR)*2,22+I)GR$(0),2
3050 FOR J=1 TO 100 :NEXT J
3060 NEXT I
3070 FOR I=1 TO NBVR*2-1
3080 PSET(I+(3-NBVR)*2,23)CHR$(127),0
3090 PSET(I+1+(3-NBVR)*2,23)GR$(0),2
3100 FOR J=1 TO 100 :NEXT J
3110 NEXT I
3120 PSET(8,23)GR$(3),4
3130 FOR J=1 TO 300 :NEXT J
3140 PSET(8,23)CHR$(127),0
3150 PSET(8,22)GR$(4),4:CR=SCREEN(9,23)
3160 FOR J=1 TO 300 :NEXT J
3170 FOR I=0 TO 3
3180 FOR J=1 TO 100 :NEXT J
3190 PSET(6+I,23)CHR$(127),0
3200 PSET(7+I,23)GR$(0),2
3210 NEXT I
3220 PSET(9,23)CHR$(CR),-(CR=43)*3
3230 PSET(7,23)GR$(3),4
3240 PSET(8,22)GR$(2),4
3250 FOR J=1 TO 100:NEXT J
3260 PSET(7,23)CHR$(127),0
3270 PSET(8,23)GR$(2),4
3300 FOR I=5 TO 0 STEP -1
3310 LOCATE 19,12,0:COLOR 2
3320 PRINT I:PLAY "DO"
3330 FOR J=1 TO 600:NEXT J
3340 NEXT I
3350 NBVR=NBVR-1
3360 TH=2:TV=23:AV=0:AH=1:V$=GR$(0)
3370 UH=23:UV=16:EH=0:EV=1:E$=GR$(1)
3380 Q1=127
3390 RETURN
```

```
4000 'FIN JEU
4010 COLOR 2:LOCATE17,9:PRINT "ENCORE"
4020 LOCATE 17,10:PRINT"UN":LOCATE17,11:PRINT"PETI
T":LOCATE17,12:PRINT"TOUR":LOCATE17,13:PRINT"(O/N)
":LOCATE17,14:INPUT REP$
4030 IF LEFT$(REP$,1)="N" THEN CLS:END
4040 IF MSC<SC THEN MSC=SC
4050 CLS:GOTO 110
5000 'PARTIE SUPPLEMENTAIRE
5010 NBVR=NBVR+1:Q=43:Q1=43
5020 FOR I=0 TO 3
5030 BOX(9+I*2,1+I*2)-(31-I*2,23-I*2)"+",3
5040 NEXT I
5050 GOTO 170
6000 'PRESENPATION DU JEU
6010 CONSOLE 0,24:SCREEN 2,0,0:CLS
6050 LINE (0,13)-(39,13)CHR$(127),4
6060 LINE (0,20)-(39,20)CHR$(127),4
6080 ATTRB1,1
6090 COLOR 3
6100 FOR I=1 TO 39 STEP 2
6110 LOCATE I,17,0:PRINT"-"
6120 NEXT I
6130 LOCATE 5,17:COLOR 2:PRINT GR$(0)
6140 LOCATE20,17:COLOR 1:PRINT GR$(0)
6150 LOCATE 10,4:COLOR 5:PRINT"STOCK-CAR"
6160 LOCATE 9,6:COLOR 1:PRINT"=========="
6170 LINE (7,20)-(15,20)CHR$(127),0
6180 LINE (7,20)-(7,24)CHR$(127),4
6190 LINE (15,20)-(15,24)CHR$(127),4
6200 ATTRB 0,0
6210 FOR I=18 TO 24 STEP 2
6220 PSET(11,I)GR$(2),3
6230 NEXT I
6240 LOCATE 0,0,0:RETURN
10000 'CREATION DU TABLEAU DE TRAVAIL
10010 FOR J=0 TO 11
10020 READ A$
10030 FOR I=0 TO 12
10040 A=VAL(MID$(A$,I+1,1))
10050 ROU(I+12,J)=A
10060 ROU(J,12-I)=A
10070 ROU(12-I,24-J)=A
10080 ROU(24-J,I+12)=A
10090 NEXT I
10100 NEXT J
10200 DATA"0000000000000"
10210 DATA"5571222222230"
10220 DATA"4400000000010"
10230 DATA"4491222223020"
```

```
10240 DATA"4400000001020"
10250 DATA"4491222302020"
10260 DATA"4400000102020"
10270 DATA"6681230202020"
10280 DATA"0000010202020"
10290 DATA"0000020202020"
10300 DATA"0000020202020"
10310 DATA"0000020202020"
10320 RETURN
```

CHAPITRE **5**

Jeux avec Joystick

5.1 CHASE

Le célèbre jeu vidéo révélé par un film récent est à votre disposition sur votre TO7. Deux joueurs s'affrontent dans un espace limité à l'écran. Les concurrents disposent d'un mobile rapide qui laisse derrière lui une traînée. Les joueurs doivent se déplacer constamment en évitant tous les obstacles constitués par les deux traînées et les limites du terrain de jeu. Il est impossible de s'arrêter et interdit de revenir sur ses pas.

Dans ces conditions, le terrain se recouvrant très rapidement d'obstacles définitifs, la collision est inévitable. Le joueur le plus habile saura provoquer la collision de l'adversaire ou retarder la sienne suffisamment.

Fonctionnement du jeu

Le RUN provoque l'apparition de la page de présentation. Une action de la touche [ENTREE] permet de

commencer. Vous pouvez choisir le nombre de parties d'une manche ; il est recommandé de prendre un nombre impair pour éviter les égalités de scores. Le départ sera alors donné par l'actionnement d'une des touches [ACTION] des manettes de jeu. Un compte à rebours partant de 5 et battant la seconde permet de ne pas être surpris par le départ. Il est à noter que le premier véhicule démarrant est choisi de façon aléatoire pour ne pas favoriser l'un des deux joueurs.

Après une collision, un message affiche la couleur du véhicule ayant percuté durant 2 secondes, puis le score est indiqué.

La partie suivante démarre grâce à un enfoncement d'une touche [ACTION].

En fin de manche, le score est affiché en grosses lettres et un message vous invite à rejouer.

Structure du programme

Le sous-programme de présentation est constitué des lignes 2000 à 2270. On constate qu'il s'agit d'une présentation animée.

Les lignes 100 à 120 dessinent le terrain de jeu et les deux bolides.

Les lignes 130 à 180 effectuent le compte à rebours.

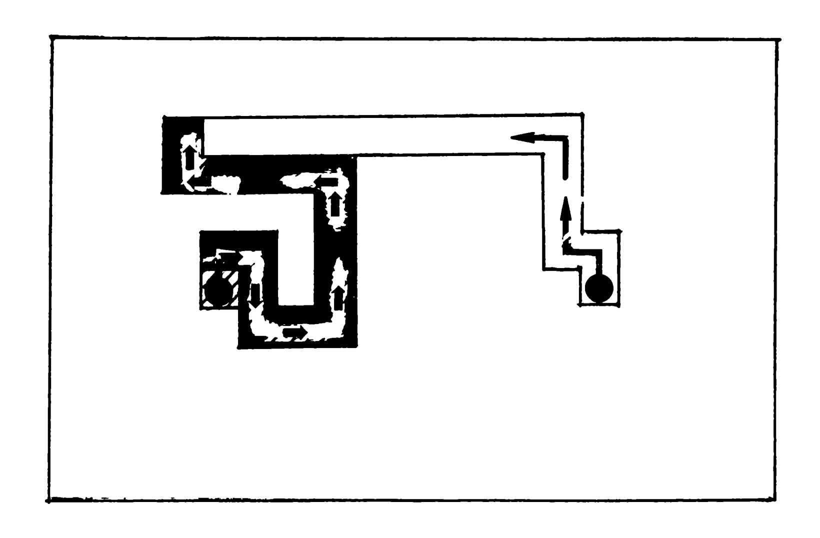
Les lignes 210 à 320 assurent le déplacement du véhicule vert (manette 0).

Les lignes 410 à 520 assurent le déplacement du véhicule rouge (manette 1).

Les lignes 1000 à 1200 affichent les résultats en cours de manche, puis le score final et enfin offrent la possibilité de rejouer.

Exemple :

Le sombre a gagné



```
10 GOSUB 2000
20 ATTRB 0,0:COLOR 2:CLS:LOCATE 0,12
30 INPUT"COMBIEN DE PARTIES PAR MANCHE";NPAR
40 PPAR=INT(ABS(NPAR))
50 NPAR=PPAR:SR=0:SV=0
60 SCREEN ,0,0
70 CLS:LOCATE 0,12,0
80 PRINT "POUR LE DEPART,ENFONCER LA TOUCHE ACTION
 D'UNE DES MANETTES DE JEU"
90 IF STRIG(0) OR STRIG(1) THEN ELSE GOTO 90
100 SCREEN ,0,0:CLS
105 BOX(0,0)-(39,24)CHR$(127),4
110 PSET(4,12)CHR$(127),1
120 PSET(35,12)CHR$(127),2
130 ATTRB 1,1:COLOR 3
140 FOR I=5 TO 0 STEP -1
150 LOCATE 17,12,0:PRINT I
160 FOR J=1 TO 500:NEXTJ
170 NEXT I
180 BOXF(100,50)-(200,150),-1
190 COLOR 2:ATTRB 0,0
200 VH=0:VV=-1:H1=35:V1=12
```

```
205 RH=0:RV=-1:H2=4:V2=12
207 IF RND>0.5 THEN GOTO 410
210 ON STICK(0) GOTO 230,300,240,300,250,300,260,3
00
220 GOTO 300
230 VH=0:VV=-1:GOTO 300
240 VH=1:VV=0:GOTO 300
250 VH=0:VV=1:GOTO 300
260 VH=-1:VV=0
300 H1=H1+VH:V1=V1+VV
310 IF SCREEN(H1,V1)=127THEN GOTO 1010
320 PSET(H1,V1)CHR$(127),2
410 ON STICK(1) GOTO 430,500,440,500,450,500,460,5
00
420 GOTO 500
430 RH=0:RV=-1:GOTO 500
440 RH=1:RV=0:GOTO 500
450 RH=0:RV=1:GOTO 500
460 RH=-1:RV=0
500 H2=H2+RH:V2=V2+RV
510 IF SCREEN(H2,V2)=127THEN GOTO 1030
520 PSET(H2,V2)CHR$(127),1:GOTO 210
1000 'FIN DE PARTIE
1010 LOCATE0,0:COLOR 2:PRINT"LE VERT A PERDU"
1020 SR=SR+1:GOTO 1050
1025 SCREEN,,0:CLS
1030 LOCATE0,0:PRINT"LE ROUGE A PERDU"
1040 SV=SV+1
1050 FOR I=1 TO 1000:NEXT I
1060 LINE (0,0)-(39,0)CHR$(127),0
1070 COLOR2:LOCATE 0,0:PRINT"ROUGE :";SR
1080 LOCATE 20,0:PRINT"VERT :";SV
1090 IF STRIG(0)ORSTRIG(1) THEN ELSE GOTO 1090
1120 NPAR=NPAR-1:IF NPAR<>0 THEN GOTO 100
1125 SCREEN,0:CLS
1130 ATTRB 1,1:LOCATE 4,4:COLOR 4:PRINT"FIN DE LA
PARTIE"
1150 COLOR 1:LOCATE 5,12:PRINT "ROUGE"
1160 LOCATE 7,16:PRINT SR
1170 COLOR 2:LOCATE 25,12:PRINT "VERT"
1180 LOCATE 25,16:PRINT SV
1190 ATTRB 0,0:LOCATE 0,23:INPUT"VOULEZ VOUS REJOU
ER";REP$
1200 IF LEFT$(REP$,1)="N"THEN CLS:END ELSE GOTO 50
2000 'DEF DU JEU
2010 SCREEN 1,0,0:CLS
2020 LINE(24,12)-(39,12)CHR$(127)
2025 COLOR 2:LOCATE 0,23,0:PRINT"APPUYER SUR ENTRE
E POUR COMMENCER"
```

```
2030 COLOR 2:LOCATE 0,23,0:PRINT"APPUYER SUR ENTRE
E POUR COMMENCER"
2040 LINE(24,12)-(39,12)CHR$(127),1
2050 PSET(11,0)CHR$(127),2:ATTRB 1,1
2060 FOR I=6 TO 4 STEP -1
2070 LINE(13,4)-(17,4)CHR$(127),1
2080 LINE(13,4)-(13,10)CHR$(127)
2090 LINE(13,10)-(39,10)CHR$(127)
2100 LOCATE 19,9:PRINT "HASE"
2110 IF INKEY$=CHR$(13)THEN GOTO 2270
2120 FOR J=1 TO 200:NEXT J
2130 NEXT I
2140 LINE(12,12)-(23,12)CHR$(127),0
2150 LINE(12,12)-(12,17)CHR$(127)
2160 LINE(11,0)-(11,12)CHR$(127)
2170 LINE(6,12)-(10,12)CHR$(127)
2180 IF INKEY$=CHR$(13)THEN GOTO 2270
2190 FOR I=1 TO 12
2200 PSET(24-I,12)CHR$(127),1
2210 PSET(11,I)CHR$(127),2
2220 FOR J=1 TO 30:NEXTJ:NEXT I
2230 FOR I=1 TO 5
2240 PSET(12,12+I)CHR$(127),1
2250 PSET(11-I,12)CHR$(127),2
2260 FOR J=1 TO 30:NEXTJ:NEXT I:GOTO 2040
2270 RETURN
```

5.2 CHENILLE

Une petite chenille se déplace sur l'écran du téléviseur à la recherche de nourriture. Ce qu'elle préfère par dessus tout, ce sont les chiffres qui surgissent tout autour d'elle, particulièrement les gros chiffres !! Malheureusement pour elle, la petite chenille a la vue très basse, et ses chances de passer sur un chiffre pour le manger sont bien faibles si quelqu'un ne vient pas à son secours pour la guider.

Vous aurez certainement à coeur d'aider cette chenille à se nourrir en la dirigeant avec le joystick, et cela d'autant plus que votre score à ce jeu est égal à la somme des chiffres mangés par la chenille.

Chaque fois que la chenille avale un chiffre, elle dépose aussitôt derrière elle, en signe de contentement, une traînée de petits cailloux dont le nombre dépend du chiffre mangé.

Toute collision avec les bords du terrain ou bien avec l'un des cailloux déposés par la chenille est fatale à celle-ci. C'est toujours de cette façon que se termine le jeu. Le joueur peut alors voir le score obtenu.

Au début du jeu, le nombre de joueurs vous est demandé. Neuf au maximum sont acceptés. Entrez ensuite les noms des joueurs (seules les dix premières lettres de chaque nom seront conservées).

Le jeu débute juste après l'introduction du dernier nom. C'est alors au premier joueur de diriger la chenille avec le joystick (joystick I) jusqu'à ce qu'une collision fatale ne mette fin à son tour de jeu.

Pressez le bouton de la manette de jeu pour voir le score réalisé par le joueur qui vient de terminer son tour de jeu. Pressez à nouveau ce bouton pour passer au joueur suivant.

Quand tous les joueurs ont joué une fois chacun, le TO7 vous propose de rejouer. Pressez soit la touche [O] soit simplement [ENTREE] pour rejouer. La touche [N] arrête le jeu définitivement.

Si vous désirez rejouer avec les mêmes joueurs, pressez simplement [ENTREE] en réponse à la question COMBIEN DE JOUEURS ?

Le meilleur score ainsi que le nom du joueur qui l'a réalisé sont gardés en mémoire. Le titre de champion vous reviendra-t-il ou bien sera-t-il décerné à l'un de vos amis ? Que le meilleur gagne !

<u>Structure du programme</u>

<u>Programme principal</u> : Lignes 10 à 80

Ligne 10, réservation de place mémoire pour un caractère défini.

Ligne 20, appel du sous-programme de présentation.

Ligne 30, appel du sous-programme d'entrée des noms de joueurs.

Ligne 40, appel du sous-programme de jeu.

Lignes 50 à 80, fin de partie classique.

<u>Sous-programme de présentation</u> : Lignes 1000 à 1330

Ligne 1000, déclaration des tableaux destinés à contenir l'un les scores des joueurs, l'autre les noms des joueurs.

Ligne 1010, la variable P$ contient les notes de la gamme chromatique, choix de l'attaque, du tempo et de la longueur des notes.

Ligne 1020, définition du caractère représentant un petit caillou.

Ligne 1030, couleur de l'écran : noir - couleur des caractères : vert - effacement de l'écran - numéro du jeu en cours à zéro.

Lignes 1040 à 1060, affichage multicolore du nom du jeu en caractères double taille et double hauteur.

Lignes 1070 à 1100, affichage de quatre chiffres en double hauteur et double largeur.

Ligne 1110, affichage de la chenille à sa position de départ.

Lignes 1120 à 1200, déplacement de la chenille sur l'écran de façon à ce qu'elle vienne toucher le chiffre 9 en haut de l'écran.

Ligne 1250, bruitage.

Lignes 1260 à 1310, initialisation du générateur de nombres aléatoires.

Ligne 1320, record et variable auxiliaire ANJO à zéro.

Ligne 1330, retour au programme principal.

Sous-programme d'entrée des noms des joueurs : Lignes 2000 à 2100

Ligne 2000, demande le nombre de joueurs.

Ligne 2010, transforme en variable numérique la réponse donnée à la question précédente.

Ligne 2020, dans le cas où seule la touche [ENTREE] a été pressée, les joueurs restent les mêmes que dans le jeu précédent.

Ligne 2030, le jeu n'accepte que neuf joueurs au maximum.

Ligne 2040, une lettre n'est pas un nombre de joueurs... elle est donc refusée.

Lignes 2050 à 2080, entrée des noms des joueurs (ligne 2050) dont seules les dix premières lettres sont conservées.

Ligne 2090, effacement de l'écran, c'est au joueur 1 de jouer (JO = 1), le nombre de parties déjà jouées est augmenté d'une unité.

Ligne 2100, retour au programme principal.

Sous-programme de jeu : Lignes 3000 à 3590

Lignes 3000 et 3010, effacement de l'écran, affichage du nom du joueur et de numéro de la partie en cours.

Ligne 3020, mise à zéro du score du joueur JO et du compteur de cailloux à déposer.

Lignes 3030 à 3100, dessin du terrain de jeu.

Ligne 3110, direction initiale de la chenille:→ et choix de sa position de départ.

Ligne 3120, lecture du joystick 1.

Ligne 3130, si le manche du joystick est en position neutre, la chenille continue d'avancer dans la direction qu'elle avait prise auparavant.

Ligne 3140, sauvegarde de l'état du joystick et des coordonnées actuelles de la chenille.

Lignes 3150 à 3160, calcul des nouvelles coordonnées de la chenille.

Ligne 3170, test si la chenille va heurter ou non soit l'un des bords du terrain soit un caillou.

Lignes 3180 et 3190, si la chenille passe sur un chiffre, le score du joueur augmente ainsi que le nombre de cailloux à déposer.

Ligne 3200, affichage de la chenille à sa nouvelle position, effacement de la chenille à son ancienne position.

Ligne 3210, pose des cailloux s'il y a lieu de le faire.

Ligne 3220, appel du sous-programme d'apparition d'un chiffre si la fonction RND prend une valeur supérieure à 0,90.

Ligne 3230, retour en 3120 pour une nouvelle lecture du joystick.

Ligne 3500, affichage de la chenille en bleu et bruitage en cas d'accident.

Ligne 3510, attend que le bouton de la manette de jeu soit pressé pour passer à la suite.

Lignes 3520 à 3560, affichage du score du joueur et du record avec modification, s'il y a lieu de le faire, de celui-ci.

Lignes 3570 à 3590, passage au joueur suivant après pression du bouton de la manette de jeu sauf si tous les joueurs viennent de jouer une fois chacun, auquel cas il y aura retour au programme principal.

Sous-programme d'apparition d'un chiffre : Lignes 4000 à 4060

Ligne 4000, choix d'un chiffre.

Ligne 4010, choix d'une position.

Lignes 4020 et 4030, refus d'une position déjà occupée soit par un caillou soit par la chenille.

Ligne 4040, affichage du chiffre.

Ligne 4060, retour au programme principal.

<u>Sous-programme utilitaire</u> : Lignes 5000 à 5060

Lignes 5000 à 5050, boucle dont les paramètres A, B, D, DH et DV sont fixés par le programme principal avant l'appel du sous-programme. Cette boucle permet de faire avancer la chenille suivie par une traînée de cailloux colorés.

Ligne 5060, retour au programme principal.

```
10 CLEAR,,1
20 GOSUB 1000
30 GOSUB 2000
40 GOSUB 3000
50 LOCATE0,24,0:INPUT"VOULEZ-VOUS REJOUER";REP$
60 IF LEFT$(REP$,1)="O" OR REP$="" THEN 30
70 IF LEFT$(REP$,1)="N" THEN CLS:END
80 GOTO 50
1000 DIM S(9),NOM$(9)
1010 P$="DODO#RERE#MIFASOSO#LALA#SI":PLAY"AOT1L4"
1020 DEFGR$(0)=0,24,60,126,126,60,24,0
1030 SCREEN2,0,0:CLS:NP=0
1040 ATTRB1,1:LOCATE11,10,0
1050 COLOR1,0:PRINT"C";:COLOR2,0:PRINT"H";:COLOR3,0:PRI
NT"E";:COLOR4,0:PRINT"N";
1060 COLOR5,0:PRINT"I";:COLOR6,0:PRINT"L";:COLOR7,0:PRI
NT"L";:COLOR1,0:PRINT"E"
1070 LOCATE17,2:COLOR5,0:PRINT"9"
1080 LOCATE5,12:COLOR3,0:PRINT"8"
1090 LOCATE30,15:COLOR2,0:PRINT"4"
1100 LOCATE34,6:COLOR7,0:PRINT"2"
1110 LOCATE3,22,0:COLOR2,0:PRINT"*"
1120 A=3:B=35:D=2:DH=2:DV=0
1130 H=3:V=22
1140 GOSUB 5000
1145 LOCATE 37,22,0:COLOR 4,0:PRINT GR$(0)
1150 A=20:B=4:D=-D:DH=0:DV=-2
1160 H=37:V=20
1170 GOSUB 5000
```

```
1175 LOCATE 37,2,0:COLOR 2,0:PRINT GR$(0)
1180 A=17:B=33:D=-D:DH=-2:DV=0
1190 H=35:V=2
1200 GOSUB 5000
1250 FORI=1TO10:PLAY P$:NEXTI
1260 COLOR2,0:ATTRB0,0:LOCATE0,24,0
1270 INPUT"ENTREZ UN NOMBRE QUELCONQUE ",N
1280 N=INT(ABS(N))
1290 FORI=1 TO N
1300 Y=RND
1310 NEXTI
1320 RECORD=0:ANJO=0
1330 RETURN
2000 CLS:INPUT"COMBIEN DE JOUEURS";NJO$
2010 NJO=VAL(NJO$)
2020 IF NJO$="" THEN NJO=ANJO:GOTO2090
2030 IF NJO>9 THEN 2000
2040 IF NJO=0 THEN 2000
2050 FORI=1 TO NJO
2060 LOCATE0,I*2:PRINT"NOM DU JOUEUR";I;:INPUT NOM$(I)
2070 IF LEN(NOM$(I))>10 THEN NOM$(I)=MID$(NOM$(I),1,10)
2080 NEXTI
2090 CLS:JO=1:NP=NP+1:ANJO=NJO
2100 RETURN
3000 CLS:COLOR2,0:LOCATE0,0,0:PRINT"JOUEUR";JO;": ";NOM
$(JO)
3010 LOCATE33,0,0:PRINT"JEU";NP
3020 S(JO)=0:E=0
3030 FORI=0TO39
3040 PSET(I,1)GR$(0),1
3050 PSET(I,24)GR$(0),1
3060 NEXTI
3070 FORJ=1TO24
3080 PSET(0,J)GR$(0),1
3090 PSET(39,J)GR$(0),1
3100 NEXTJ
3110 A=3:H=10:V=10
3120 B=STICK(1)
3130 IF B=0 THEN B=A
3140 A=B:HP=H:VP=V
3150 H=H+(B=7)-(B=3)
3160 V=V+(B=1)-(B=5)
3170 IF POINT(H*8+4,V*8+4)=1 THEN 3500
3180 C=SCREEN(H,V)
3190 IF C>48 THEN S(JO)=S(JO)+C-48:E=E+C-48
3200 PSET(H,V)"*",2:PSET(HP,VP)" "
3210 IF E>0 THEN PSET(HP,VP)GR$(0),1:E=E-1
3220 IF RND>0.90 THEN GOSUB 4000
3230 GOTO3120
3500 PSET(HP,VP)"*",4:FORI=1 TO 10: PLAY P$:NEXT I
3510 IF STRIG(1)<>-1 THEN 3510
```

```
3520 CLS:COLOR2,0
3530 LOCATE10,5,0:PRINT"JOUEUR";JO;": ";NOM$(JO)
3540 LOCATE10,10,0:PRINT"SCORE: ";S(JO)
3550 IF S(JO)>RECORD THEN LOCATE10,15,0:COLOR7,0:PRINT"
NOUVEAU RECORD: ";S(JO):RECORD=S(JO):CHAMP$=NOM$(JO):GO
TO3570
3560 LOCATE10,15,0:PRINT"RECORD TOUJOURS DETENU PAR ":L
OCATE10,17,0:COLOR7,0:PRINT CHAMP$;:COLOR2,0:PRINT" AVE
C";:COLOR7,0:PRINT RECORD
3570 COLOR2,0:JO=JO+1:IF JO>NJO THEN RETURN
3580 IF STRIG(1)<>-1 THEN 3580
3590 GOTO 3000
4000 Z$=STR$(INT(RND*8+2))
4010 X=INT(RND*38):Y=INT(RND*23+1)
4020 W=POINT(X*8+4,Y*8+4)
4030 IF W=1 OR W=2 THEN 4010
4040 LOCATEX,Y,0:COLOR7,0:PRINT Z$
4050 COLOR2,0
4060 RETURN
5000 FORI=A TO B STEP D
5010 C=INT(RND*7+1)
5020 H=H+DH:V=V+DV
5030 COLOR2,0:LOCATEH,V:PRINT"*":PLAY"DO"
5040 COLORC,0:LOCATEH-DH,V-DV,0:PRINT GR$(0)
5050 NEXTI
5060 RETURN
```

5.3 LETTRIVORE

Les vingt-six lettres de l'alphabet apparaissent dans le désordre le plus total sur l'écran de votre téléviseur. Le but du jeu est de "manger" le plus rapidement possible ces lettres en respectant l'ordre alphabétique.

Pour manger une lettre, il suffit de faire passer la chenille, que vous dirigez avec le joystick, sur cette lettre.

Toute collision avec un obstacle représenté par un petit carré rouge ou bien avec une lettre dont ce n'est pas encore le tour d'être mangée, entraîne l'arrêt du jeu.

Le chronomètre, situé dans le coin supérieur gauche de l'écran, indique en permanence le temps écoulé depuis le début du jeu. En cas de victoire (ce qui, il faut l'avouer, est assez rare car LETTRIVORE est un jeu difficile), votre temps vous est donné. Selon les cas, le record est modifié ou non.

Neuf niveaux de jeu (de 0 à 9, 0 étant le niveau le plus difficile) vous permettront de vous entraîner avec une chenille qui va lentement (niveau 9) puis avec une chenille ultra-rapide (niveau 0).

En fin de partie, pressez soit la touche [O] soit la touche [ENTREE] pour rejouer. La touche [N] arrête définitivement le jeu.

Structure du programme

Programme principal : lignes 10 à 70

Ligne 10, appel du sous-programme de présentation.

Ligne 20, appel du sous-programme d'affichage.

Ligne 30, appel du sous-programme de jeu.

Lignes 40 à 70, fin de partie classique

Sous-progamme de jeu : lignes 1000 à 1550

Ligne 1000, place la chenille au centre de l'écran - la lettre A doit être la première lettre mangée (65 est le code ASCII de A) - la chenille se déplace automatiquement de gauche à droite si le joueur ne réagit pas assez vite au début du jeu (DH = 1 ; DV = 0).

Ligne 1010, garde en mémoire la position actuelle de la chenille - incrémente le "chronomètre" T.

Ligne 1020, branchement en fonction de la lecture du joystick 1.

Lignes 1030 à 1070, différentes possibilités selon la position du manche du joystick (en 1040 : déplacement vertical vers le haut, 1070 : déplacement horizontal vers la gauche).

Ligne 1100, calcul de la nouvelle position de la chenille.

Lignes 1110 à 1120, la chenille réapparaît du côté opposé au côté par où elle vient de sortir de l'écran.

Ligne 1130, lecture de l'écran - la chenille peut continuer à avancer normalement si rien n'est décelé (obstacle ou lettre).

Ligne 1140, une collision avec une lettre dont ce n'est pas le tour d'être mangée arrête le jeu.

Ligne 1150, le score est incrémenté d'une unité - bip sonore.

Ligne 1160, on passe à la lettre suivante - si toutes les lettres ont été mangées, le jeu s'arrête.

Ligne 1170, on enlève la chenille de son ancienne position et on la met à sa nouvelle place.

Ligne 1180, boucle de temporisation dont la durée est fonction du niveau de jeu (N) choisi par le joueur - retour au début du sous-programme de jeu pour un nouveau déplacement.

Lignes 1200 à 1230, le joueur a perdu.

Lignes 1500 à 1550, le joueur a gagné - modification, s'il y a lieu, du record.

Sous-programme d'affichage : lignes 2000 à 2190

Ligne 2000, effacement de l'écran - introduction du niveau de jeu (ce niveau doit être inférieur ou égal à 9).

Ligne 2005, effacement de l'écran - mise du score et du chronomètre à zéro.

Ligne 2010, affichage de la chenille au centre de l'écran.

Ligne 2020, initialisation des variables de position de la chenille.

Lignes 2030 à 2070, placement des vingt-six lettres de l'alphabet sur l'écran en s'assurant que deux lettres ne peuvent pas se superposer (ligne 2050).

Lignes 2080 à 2120, placement des vingt-six obstacles rouges sur l'écran.

Ligne 2130, affichage du temps et du niveau.

Lignes 2140 à 2180, compte à rebours au début du jeu.

Ligne 2190, retour au programme principal.

Sous-programme de présentation : lignes 3000 à 3340

Ligne 3000, effacement de l'écran - couleur de l'écran : noir - couleur des caractères : vert.

déclaration du tableau contenant les records réalisés pour chaque niveau de jeu.

Ligne 3010, la variable chaîne P$ contient les notes de la gamme chromatique.

Ligne 3020, choix de l'attaque, tempo, durée et octave pour les notes.

Lignes 3030 à 3060, affichage en couleurs du nom du jeu.

Lignes 3070 à 3100, affichage de quatre lettres.

Lignes 3120 à 3230, déplacement de la chenille.

Lignes 3240 à 3260, bruitage exprimant la satisfaction de la chenille après que celle-ci ait mangé la lettre A.

Lignes 3270 à 3320, initialisation du générateur de nombres aléatoires.

Ligne 3330, initialisation du tableau des records.

Ligne 3340, retour au programme principal.

```
  10  GOSUB  3000
  20  GOSUB  2000
  30  GOSUB  1000
  40  LOCATE0, 23, 0  INPUT "VOULEZ VOUS REJOUER" , REP$
  50  IF LEFT$ (REP$, 1) = " O " OR REP$ = " " THEN 20
  60  IF LEFT$ (REP$, 1) = " N " THEN CLS   END
  70  GOTO 40
1000  PSET (15,12) " * "   P = 65   B = 3   DH = 1   DV = 0
1010  AH = H   AV = V   T = T+1
1020  ON STICK (1) GOTO 1040, 1100, 1050, 1100, 1060, 1100,
      1070, 1100
1030  GOTO  1100
```

```
1040  DV=-1  DH=0  GOTO1100
1050  DH=1  DV=0  GOTO1100
1060  DV=1  DV=0  GOTO1100
1070  DH=-1  DV=0  GOTO1100
1100  H=H+DH  V=V+DV
1110  IF  H=39  OR  H=0  THEN  H=ABS(H-38)
1120  IF  V=24  OR  V=0  THEN  V=ABS(V-23)
1130  C=SCREEN(H,V) IF  C=32  OR  C=42  THEN 1170
1140  IF  C<>P  THEN  1200
1150  S=S+1  PLAY  P$
1160  P=P+1  IF  P=91  THEN  1500
1170  PSET(AH,AV)"  " PSET(H,V)"*" LOCATE7,0,0 PRINT
1180  FORI=1  TO  N*TU  NEXT  I B$=A$  GOTO  1010
1200  FORI=1  TO  10  PLAY  P$  NEXTI  PSET(H,V)"
1210  LOCATE0,0,0  PRINT"VOUS AVEZ PERDU .."
1220  LOCATE0,5,0  PRINT"VOUS AVEZ MANGE" ; S ;"LETTRES"
1230  GOTO  1550
1500  FORI=1  TO  10  PLAY  P$  NEXTI  PSET(H,V)"  "
1510  LOCATE0,0,0  PRINT"VOUS AVEZ GAGNE EN" ;T ;"SEC ONDES"
1520  LOCATE0,5,0
1530  IF T<RECORD(N)  THEN  PRINT"LE RECORD AU NIVEAU" ;N ;"EST MAINTENANT
      " ;T$RECORD(N)=T RETURN
1540  PRINT"LE RECORD AU NIVEAU" ;N ;"EST TOUJOURS     " ;RECORD(N)
1550  RETURN
2000  CLS  INPUT"NIVEAU(0-9)" ;N N=INT(ABS(N))  IF  N>9
      THEN  2000
2005  CLS S=0 T=0
2010  LOCATE15,12,0  PRINT"*"
2020  H=15  V=12  COLOR  7,0
2030  FORI=65  TO  90
2040  X=INT(RND*38+1)  Y=INT(RND*23+1)
2050  IF  SCREEN(X,Y)<>32  THEN  2040
2060  LOCATEX,Y,0 PRINT  CHE$(I)
2070  NEXTI
2080  COLOR1,0:FORI=1  TO  26
2090  X=INT(RND*38+1) Y=INT(RND*23+1)
2100  IF SCREEN(X,Y)<>32  THEN  2090
2110  LOCATEX,Y,0 PRINT  CHR$(127)
2120  NEXTI
2130  COLOR2,0  LOCATE0,0,0  PRINT"TEMPS     "  T  LOCATE3
      0,0,0 PRINT"NIVEAU    " N
2140  FORI=10  TO  1  STEP-1
2150  LOCATE17,0,0 PRINT  I
2160  PLAY  P$ FORJ=1  TO  100  NEXTJ
2170  NEXTI
2180  LOCATE17,0,0  PRINT"
2190  RETURN
3000  CLS SCREEN2,0,0 DIM  RECORD(9)
```

```
3010  P$="SILA•LASO•SOFA•FAMIRE•REDO•DO"
3020  PLAY"AOT1L404"
3030  ATTRB1,1 LOCATE11,10,0
3040  COLOR1,0 PRINT"L" ; COLOR2,0 PRINT"E" , COLOR3
      0 PRINT"T" ; COLOR4,0 PRINT"T" ,
3050  COLOR5,0 PRINT"R" ; COLOR6,0 PRINT"I" ; COLOR7
      0 PRINT"V" ; COLOR1,0 PRINT"0" ;
3060  COLOR2,0 PRINT"R" ; COLOR3,0 PRINT"E"
3070  LOCATE17,2,0 COLOR5,0 PRINT"A"
3080  LOCATE5,12,0 COLOR3,0 PRINT"B"
3090  LOCATE30,15,0 COLOR2,0 PRINT"Z"
3100  LOCATE34,6,0 COLOR7,0 PRINT"P"
3110  PLAY  P$ COLOR2,0
3120  FORI=0 TO,36
3130  LOCATE1,22,0 PRINT"•"
3140  LOCATE1,22,0 PRINT" "
3150  NEXTI
3160  FORI=22  TO  2  STEP-1
3170  LOCATE36,1,0 PRINT"•"
3180  LOCATE36,I,0 PRINT" "
3190  NEXTI
3200  FORI=36  TO  17  STEP-1
3210  LOCATEI,2,0 PRINT"•"
3220  LOCATEI,2,0 PRINT" "
3230  NEXTI
3240  FORI=1  TO  10
3250  PLAY  P$
3260  NEXTI
3270  LOCATE0,23,0 ATTRB0,0
3280  INPUT"ENTREZ UN NOMBRE QUELCONQUE",N
3290  N=INT(ABS(N))
3300  FORI=1  TO  N
3310  Y=RND
3320  NEXTI
3330  FORI=0  TO  9  RECORD(I)=10000  NEXTI
3340  RETURN
```

5.4 MUR

Sans le mur de briques, une collection de jeux vidéo n'est pas complète. Chacun connaît ce jeu qui a fait la fortune et les débuts d'une des plus grandes maisons de jeux vidéo. La version proposée ici, bien que simple, présente une rapidité tout à fait honnête pour un jeu BASIC et permettra de s'amuser durant les longues soirées d'hiver moins propices aux sports de plein air que l'été.

Le mur est constitué de 7 rangées de briques et un point est attribué à chaque brique détruite. A la fin de chaque partie, c'est-à-dire lorsque la balle sort du jeu, le score et éventuellement le record sont affichés sur la gauche de l'écran. La balle prend une position aléatoire à chaque début de partie. Pour commencer une partie, il suffit d'appuyer sur la touche [ACTION] de la manette 0.

Si vous possédez des manettes de jeux, empressez-vous de rentrer ce programme qui ne déçoit jamais.

Structure du programme

Les seuls sous-programmes sont l'éternelle partie consacrée à la présentation (lignes 3000 à 3190) et le dessin du terrain de jeu (lignes 1000 à 1210).

La ligne 215 oriente le programme suivant la position de la manette de jeu.

Les lignes 230 et 240 assurent un déplacement vers la gauche de la raquette mais l'empêchent de sortir du terrain.

La ligne 250 assure quant à elle le déplacement vers la droite.

Le déplacement vers la droite est ensuite géré par l'ensemble d'instructions contenues dans les lignes 310 à 370.

Le rebond se fait dans le sens vertical si la paroi est bleue ou si des briques sont touchées et dans le sens horizontal si la paroi est blanche.

Les lignes 500 à 710 permettent d'afficher un message, puis un autre, à intervalles de temps réguliers. Ces deux messages indiquent qu'il faut appuyer

sur la touche [S] pour sortir du programme (lignes 570 à 600) ou sur la touche [ACTION] pour commencer une autre partie (lignes 510 à 550).

Ce programme est court, il est donc facile à rentrer sans erreur ; nous conseillons aux débutants d'étudier puis de faire fonctionner ce type de programme avant de passer à des programmes plus longs.

```
10 CLEAR,,1:MS=100:GOSUB 3000
20 DEFGR$(0)=0,0,60,60,60,60,0,0
30 SC=0
40 GOSUB 1000
50 R=20:AH=1:AV=-1
60 LOCATE 0,24,0:COLOR 2
65 LINE (0,24)-(39,24)CHR$(127),0:COLOR 2:LOCATE 0
,24,0
70 PRINT"APPUYER SUR LA TOUCHE ACTION";CHR$(11)
80 IF STRIG(0)THEN ELSE GOTO 80
90 LINE (0,24)-(39,24)CHR$(127),0
200 'JEU
210 IF SC=189 THEN GOTO 700
215 ON STICK(0)GOTO 220,230,230,230,220,250,250,25
0
220 GOTO 300
230 IF R<32 THEN PSET(R-1,23)CHR$(127),0:R=R+1:PSE
T(R+1,23)CHR$(127),4:GOTO 300
240 GOTO 300
250 IF R>8 THEN PSET(R+1,23)CHR$(127),0:R=R-1:PSET
(R-1,23)CHR$(127),4
300 V=BH:W=BV:BH=V+AH:BV=W+AV
310 ON POINT(BH*8,BV*8) GOTO 350,350,350,360,350,3
50,370
320 PSET(V,W)CHR$(127),0:PSET(BH,BV)GR$(0),4:IF BV
>22 THEN GOTO 500 ELSE IF BH<>14 THEN GOTO 210
330 IF AV<0 AND BH>9 AND BH<31 THEN V=BH:BH=BH+INT
(RND*2):PSET(V,BV)CHR$(127),0:PSET(BH,BV)GR$(0),4
340 GOTO 210
350 SC=SC+1:PSET(V,W)CHR$(127),0:PSET(BH,BV)GR$(0)
,4:AV=-SGN(AV):GOTO 210
360 BH=V:BV=W:AV=-SGN(AV):GOTO 210
370 BH=V:BV=W:AH=-SGN(AH):GOTO 210
500 LOCATE 0,10,0:COLOR 2:PRINT SC
510 FOR I=1 TO 20
520 LOCATE 0,24:COLOR2:PRINT"APPUYER SUR ACTION PO
UR REJOUER";CHR$(11)
```

```
530 IF STRIG(0) THEN GOTO 30
540 IF INKEY$="S" THEN CLS:END
550 NEXT I
560 FOR I=1 TO 20
570 LOCATE 0,24:COLOR2:PRINT"APPUYER SUR   S    PO
UR STOPPER";CHR$(11)
580 IF STRIG(0) THEN GOTO 30
590 IF INKEY$="S" THEN CLS:END
600 NEXT I
610 GOTO 510
700 LOCATE 0,10,0:PRINT SC
710 GOTO 50
1000 'TERRAIN
1010 SCREEN ,0,0:CLS
1020 BOXF(6,0)-(34,24)CHR$(127),0
1030 LINE(6,0)-(34,0)CHR$(127),4
1040 LINE(6,1)-(6,23)CHR$(127),7
1050 LINE(34,1)-(34,23)CHR$(127),7
1060 LINE(7,1)-(33,1)CHR$(127),3
1070 LINE(7,2)-(33,2)CHR$(127),6
1080 LINE(7,6)-(33,6)CHR$(127),2
1090 LINE(7,7)-(33,7)CHR$(127),1
1100 BOXF(7,3)-(33,5)CHR$(127),5
1110 LINE(7,6)-(33,6)CHR$(127),2
1120 LINE(7,7)-(33,7)CHR$(127),1
1130 LINE(19,23)-(21,23)CHR$(127),4
1140 BH=8+INT(RND*24):BV=9+INT(RND*10)
1150 PSET(BH,BV)GR$(0),4
1160 LOCATE 0,9,0:COLOR 2:PRINT "SCORE"
1170 LOCATE 0,10,0:PRINT SC
1180 LOCATE 0,15,0:PRINT "HIGHT"
1190 IF MS<SC THEN MS=SC
1200 LOCATE 0,16,0:PRINT MS
1210 RETURN
3000 'PRESENTATION
3010 GOSUB 1000:BOXF(0,0)-(5,24)CHR$(127),0
3120 ATTRB 1,1:COLOR 1
3140 LOCATE 12,11,0:PRINT"MUR DE"
3150 LOCATE 14,16,0:PRINT"BRIQUES"
3160 ATTRB 0,0:COLOR 2
3170 LOCATE 0,24:PRINT"APPUYER SUR ENTREE POUR COM
MENCER";CHR$(11)
3180 IF INKEY$<>CHR$(13)THEN GOTO 3180
3190 RETURN
```

5.5 PICKMAN

Le PICKMAN est un jeu bien connu de tous les amateurs de vidéo. Le PICKMAN, pour se nourrir et protéger l'humanité d'horribles crétures, est condamné à dévorer sans relâche les oeufs qu'ont pondu un couple de monstres à l'intérieur d'un labyrinthe. Bien sûr, les monstres n'aiment pas ça du tout et de plus, ils sont prêts à dévorer n'importe quelle créature surtout si celle-ci s'introduit dans leur demeure. Dans ce sombre labyrinthe, les monstres reconnaissent les intrus à l'odeur, ces odeurs arrivent même parfois à contourner les murs et donc, le PICKMAN a toutes les peines du monde à trouver des endroits où il pourra se reposer et réfléchir au moyen de manger le plus d'oeufs possibles. Lorsqu'il n'y a plus d'oeufs dans le labyrinthe, les monstres mettent au monde une nouvelle couvée que le PICKMAN doit à nouveau détruire. Tout comme le PICKMAN, serez-vous capable de tenir indéfiniment ce rythme infernal ? Votre TO7 est en mesure de vous mettre à l'épreuve ; il ne vous reste donc plus qu'à essayer !

Fonctionnement du programme

Après avoir vérifié plusieurs fois que vous avez correctement transcrit le programme sur votre TO7, et effectué la commande RUN, vous avez le choix entre plusieurs labyrinthes ; vous écrirez donc le numéro choisi suivi de [ENTREE] et le jeu commencera.

Le déplacement s'effectue dans les sens horizontaux et verticaux à l'exclusion des diagonales à l'aide de la manette 0.

Lorsque la partie est finie, vous pouvez recommencer en appuyant sur la touche [0] puis [ENTREE].

Votre score, ainsi que le meilleur score, sont indiqués sur la gauche de l'écran.

Structure du programme

Le programme principal s'étend de la ligne 10 à la ligne 1070 et est composé de 5 blocs.

Bloc d'initialisation : lignes 10 à 190

Comme son nom l'indique, le rôle de ce bloc est de préparer l'ensemble des variables ainsi que l'écran pour un déroulement correct du reste du programme. Toutes les variables sauf celles commençant par S sont déclarées comme entières, ceci dans le but d'augmenter légèrement la vitesse d'exécution.

Tous les sous-programmes étant appelés par ce bloc, nous les décrivons ci-dessous.

Sous-programme de définition des caractères graphiques : lignes 5000 à 5080

Ce sous-programme se charge de définir les 6 caractères graphiques utilisés dans le jeu. Il s'agit

- des 4 directions possibles du PICKMAN
- des monstres (un seul caractère)
- des oeufs (un seul caractère)

Sous-programme de présentation

Ce sous-programme affiche de façon classique la page de présentation et interroge le joueur sur le numéro du labyrinthe dans lequel il désire jouer (lignes 1270 à 1280).

Sous-programme d'affichage des scores initiaux : lignes 5100 à 5160

Ce sous-programme affiche sur la gauche de l'écran les mots "score" et "record", et en-dessous les valeurs initiales respectives.

Sous-programme de dessin du terrain de jeu : lignes 5200 à 5850

Le labyrinthe et les oeufs sont dessinés grâce aux lignes 5210 à 5260. Les lignes 5280 et suivantes contiennent la forme des 3 labyrinthes disponibles. Il est possible de modifier cette forme et même d'ajouter des formes supplémentaires.

Prenons le premier labyrinthe. Sa forme est contenue sous forme "DATA" dans les lignes 5280 à 5450. Les zéros représentent l'absence de mur et les 1 un élément de mur. Les seules conditions à respecter sont de laisser libres les 3 cases correspondant aux deux monstres et au PICKMAN lors du début de la partie et d'avoir un nombre total de zéros égal à 233.

L'augmentation du nombre de labyrinthes se fait en ajoutant à partir de la ligne 5850 des lignes identiques aux lignes 5280 à 5840 (les conditions ci-dessus devant être respectées) et en modifiant la ligne 1280. Cette dernière modification consiste à changer le nombre 4 par N + 1 où N est le nouveau nombre de labyrinthes.

Sous-programme d'initialisation des variables : lignes 4100 à 4140

Dans cette partie, les positions de départ des trois programmes sont initialisées aini que le nombre d'oeufs.

Sous-programme d'affichage des positions de départ

Faisant suite au précédent, ce sous-programme place les deux monstres et le PICKMAN avec leurs couleurs respectives sur le terrain de jeu.

Bloc de déplacement du PICKMAN : lignes 200 à 320

Après avoir testé la valeur de la manette de jeu, le programme est aiguillé vers la ligne correspondant au déplacement. La ligne 320 permet d'accélérer la vitesse de déplacement du PICKMAN, il est possible d'optimiser cette vitesse en modifiant la valeur 0,5.

Bloc de déplacement des monstres : lignes 500 à 700

Les deux monstres se meuvent tour à tour dans un ordre aléatoire. Connaissant les coordonnées du PICKMAN et les leurs, il est facile de provoquer un déplacement qui tend à rapprocher les monstres du PICKMAN. Il faut d'autre part restituer un oeuf si la case que quitte le monstre en contenait un antérieurement. L'ensemble de ces fonctions est réalisé par ce bloc. La ligne 700 teste si les monstres ont réussi à manger PICKMAN.

Bloc "mange" : lignes 800 à 920

Ce sous-programme traite le cas où le PICKMAN vient d'être rattrapé par un des monstres. La case où se trouvait le PICKMAN est noircie puis les constantes de départ réinitialisées. S'il ne reste plus de PICKMAN dans la réserve, le jeu est terminé. On se branche donc à la ligne 1000 (830).

Bloc de fin de jeu

Ce bloc est très classique et permet au joueur de recommencer une autre partie ou de sortir du programme (ligne 1030).

Lorsque la version "clavier" de ce jeu est utilisée, tout ce qui vient d'être dit reste valable sauf pour les lignes 200 à 320 qui sont purement et simplement remplacées par celles décrites au chapitre 4. Il faut cependant souligner que la manette de jeu permet un déroulement bien plus rapide de programme que le clavier.

Après ces explications, il ne vous reste plus maintenant qu'à rentrer votre programme et à passer de longues minutes à résister aux monstres.

Les trois labyrinthes proposés ont volontairement été choisis pour présenter des niveaux de difficulté différents. Le 3 est le plus simple, le 1 demande un peu d'entraînement, quant au 2, il est très difficile d'avaler tous les oeufs avec le même PICKMAN.

Pour changer de labyrinthe, on est obligé de refaire un RUN.

```
0 'PICKMAN
10 CLEAR,,7:DEFINT A-R,T-Z
20 SCREEN,0,0:CLS:SCORM=3000
30 GOSUB 5000
40 GOSUB 1200
50 GOSUB 5100
60 SCORE=0:RES=3
70 GOSUB 5200
80 GOSUB 4100
90 GOSUB 4000
100 COLOR 2
110 LOCATE 31,12,0
```

```
120 IF RES=2 THEN PRINT GR$(2);"  "
130 IF RES=3 THEN PRINT GR$(2);" ";GR$(2)
140 IF RES=1 THEN PRINT "   "
190 I=1:V=PH:W=PV
200 'DEPLACEMENT DU PICKMAN
210 V=PH:W=PV:ON STICK(0) GOTO 230,510,240,510,250
,510,260,510
220 GOTO 510
230 PAC$=GR$(5):PV=PV-1:GOTO 270
240 PAC$=GR$(3):PH=PH+1:GOTO 270
250 PAC$=GR$(4):PV=PV+1:GOTO 270
260 PAC$=GR$(2):PH=PH-1
270 YL=POINT(PH*8+3,PV*8+3)
280 OEUF=OEUF+(YL=5):IF YL=4 OR YL=1 THEN PH=V:PV=
W
290 PSET(V,W)CHR$(127),0:PSET(PH,PV)PAC$,2
300 SCORE=SCORE-17*(YL=5):LOCATE 0,13,0:PRINT SCOR
E
310 IF OEUF=0THEN GOTO 1500
320 IF RND>0.5 THEN GOTO 210
500 'DEPLACEMENT DES MONSTRES
510 I=INT(2*RND)+1:O=0
530 A=(PH<MH(I))-(PH>MH(I)):B=(PV<MV(I))-(PV>MV(I)
)
580 X=MH(I)+A:Y=MV(I)+B
600 IF POINT(X*8,Y*8+7)<1 THEN GOTO 660
610 R=RND:ON INT(2*RND)GOTO 640
620 X=X+(A=0)*((R<0.5)-(R>0.5))
630 IF POINT(X*8,MV(I)*8+7)<1 THEN Y=MV(I):GOTO 66
0
640 Y=Y+(B=0)*((R<0.5)-(R>0.5))
650 IF POINT(MH(I)*8,Y*8+7)>0THENGOTO210 ELSE X=MH
(I)
660 O=-(POINT(X*8+3,Y*8+4)=5):PSET(X,Y)GR$(6),1
670 IF OE(I)=1 THEN PSET(MH(I),MV(I))GR$(1),5 ELSE
 PSET(MH(I),MV(I))CHR$(127),0
690 MH(I)=X:MV(I)=Y:OE(I)=O
700 IF(MH(1)=PH)*(MV(1)=PV)+(MH(2)=PH)*(MV(2)=PV)=
0 THEN GOTO 210
800 'MANGE
810 PSET(PH,PV)GR$(6),1
820 RES=RES-1
830 IF RES=0 THEN GOTO 1000
840 FOR J=1 TO 2
850 IF OE(J)=1 THEN PSET(MH(J),MV(J))GR$(1),5 ELSE
 PSET(MH(J),MV(J))CHR$(127),0
860 NEXT J
870 PH=21:PV=14:MH(1)=20:MV(1)=4:MH(2)=10:MV(2)=11
880 FOR J=1 TO 2
```

```
890 IF POINT(MH(J)*8+3,MV(J)*8+3)=5 THEN OE(J)=1 E
LSE EO(J)=0
900 NEXT J
920 GOTO 90
1000 'FIN JEUX
1010 COLOR 1,0 :LOCATE 0,0,0
1020 INPUT "VOULEZ VOUS REJOUER";REP$
1030 IF ASC(REP$)<>79 THEN CLS:COLOR 2:END
1040 IF SCORE>SCORM THEN SCORM=SCORE
1050 BOXF(9,3)-(28,22)CHR$(127),0
1060 LINE(0,0)-(39,0)CHR$(127),0
1070 GOTO 50
1200 'PRESENTATION
1210 CLS:ATTRB 1,1
1220 COLOR 1,0
1230 LOCATE 12,10:PRINT"PICKMAN",GR$(6)
1240 COLOR 4:LOCATE 9,14
1250 PRINT "=========="
1260 COLOR 2:LOCATE 5,10 :PRINT GR$(3)
1270 ATTRB 0,0:LOCATE 5,23:INPUT"QUEL LABYRINTHE D
ESIREZ VOUS";NL$
1280 NL=VAL(NL$) MOD4
1290 CLS
1300 RETURN
1500 'FIN TAB
1510 BOXF(9,3)-(28,22)CHR$(127),0
1520 GOTO 70
4000 'RAJOUT 1
4010 PSET(21,14)GR$(2),2,0
4020 PSET(20,4)GR$(6),1,0
4030 PSET(10,11)GR$(6),1,0
4040 RETURN
4100 'CONFIGURATION DE DEPART
4110 PH=21:PV=14
4120 MH(1)=20:MV(1)=4:MH(2)=10:MV(2)=11
4130 OEUF=232:OE(1)=1:OE(2)=1
4140 RETURN
5000 'DEFINITION DES CARACTERES
5020 DEFGR$(1)=0,0,0,24,24,0,0,0
5030 DEFGR$(2)=60,254,63,31,15,63,254,60
5040 DEFGR$(3)=60,127,252,240,248,252,127,60
5050 DEFGR$(4)=60,126,255,255,247,231,66,66
5060 DEFGR$(5)=66,66,231,247,255,255,126,60
5070 DEFGR$(6)=56,124,214,214,254,254,170,170
5080 RETURN
5100 'RAJOUT 2
5110 COLOR 2,0
5120 LOCATE 1,12,0:PRINT "SCORE"
5130 LOCATE 1,13,0:PRINT "000","
5140 LOCATE 1,5,0:PRINT "RECORD"
5150 LOCATE 0,6,0:PRINT SCORM
```

```
5160 RETURN
5200 'DEF DU LABYRINTHE
5210 BOX(9,3)-(28,22)CHR$(127),4
5211 RESTORE:IF NL=1 THEN GOTO 5220
5212 FOR I=1 TO 18*(NL-1):READ E$:NEXT I
5220 FOR J=4 TO 21
5230 READ E$
5240 FOR I=1 TO 18
5250 IF MID$(E$,I,1)="1"THEN PSET(I+9,J)CHR$(127),
4 ELSE PSET(I+9,J)GR$(1),5
5260 NEXT I
5270 NEXT J
5280 DATA"000000000001110000"
5290 DATA"000110111000100001 1"
5300 DATA"010010100001100110"
5310 DATA"010000000000000000"
5320 DATA"011100111000000000"
5330 DATA"000100000000111111"
5340 DATA"000110000000010000"
5350 DATA"000000001000010010"
5360 DATA"110000001010000010"
5370 DATA"011000000011101110"
5380 DATA"000011100000101000"
5390 DATA"000010000111100000"
5400 DATA"001110000000000110"
5410 DATA"000010000000100000"
5420 DATA"010010001111101000"
5430 DATA"010000011000001000"
5440 DATA"011111000001111001"
5450 DATA"000000000000100000"
5460 '2EME LAB.
5470 DATA 000000000000000000
5480 DATA 001110011110011100
5490 DATA 000000000000000000
5500 DATA 010000001100000010
5510 DATA 010100101101001010
5520 DATA 000100101101001000
5530 DATA 000100101101001000
5540 DATA 010000000000000010
5550 DATA 010111100001111010
5560 DATA 010111100001111010
5570 DATA 010000000000000010
5580 DATA 000100101100001000
5590 DATA 000100101101001000
5600 DATA 010100101101001010
5610 DATA 010000001100000010
5620 DATA 000000000000000000
5630 DATA 001110011110011100
5640 DATA 000000000000000000
5650 '3EME LAB.
```

```
5670 DATA 000000000000000000
5680 DATA 000000000000000000
5690 DATA 001111110011111100
5700 DATA 001000000000000100
5710 DATA 001011110011110100
5720 DATA 001010000000010100
5730 DATA 001010111111010100
5740 DATA 001010100001010100
5750 DATA 000000100001000000
5760 DATA 000000100001000000
5770 DATA 001010100001010100
5780 DATA 001010111111010100
5790 DATA 001010000000010100
5800 DATA 001011110011110100
5810 DATA 001000000000000100
5820 DATA 001111110011111100
5830 DATA 000000000000000000
5840 DATA 000000000000000000
5850 RETURN
```

5.6 SQUASH

Ce jeu dérivé du fameux mur de briques n'est pas basé sur l'élimination de cibles à l'aide de la balle mais sur l'anticipation des mouvements. En effet, la salle de jeu n'est pas vide et contient en son milieu deux murs de petites dimensions qui provoquent une multitude de rebonds et le joueur pris de vitesse se trouve-bien souvent-très dérouté. La difficulté est en fait supérieure à celle du mur de briques.

Le nombre de points gagnés est fonction du temps passé sans perdre la balle.

<u>Fonctionnement du jeu</u>

Après avoir appuyé sur la touche [ENTREE] lorsque la page de présentation était affichée, le terrain de jeu apparaît ainsi que la balle dont la position est aléatoire. Il faut alors enfoncer la touche [ACTION] de la manette de jeu 0 ; la partie commence alors et continue jusqu'à ce que le joueur laisse sortir la balle du terrain.

Structure du programme

Lignes 230 et 240, provoquent le déplacement de la raquette vers la gauche alors que le mouvement contraire est pris en charge par la ligne 250.

Lignes 310 et 370, assurent les mouvements de la balle.

Lignes 500 à 610, permettent de rejouer ou bien de sortir du programme.

Lignes 1000 à 1200, dessinent le terrain de jeu.

Lignes 3000 à 3190, constituent la partie présentation du programme.

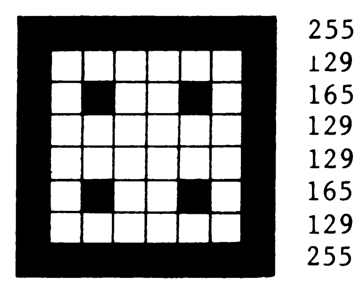
```
10 CLEAR,,1:MS=100:GOSUB 3000
20 DEFGR$(0)=0,0,60,60,60,60,0,0
30 'DEBUT
40 GOSUB 1000
50 R=20:AH=1:AV=-1
60 LOCATE 0,24,0:COLOR 2
65 LINE (0,24)-(39,24)CHR$(127),0:COLOR 2:LOCATE 0
,24,0
70 PRINT"APPUYER SUR LA TOUCHE ACTION";CHR$(11)
80 IF STRIG(0)THEN ELSE GOTO 80
90 LINE (0,24)-(39,24)CHR$(127),0
200 'JEU
210 SC=SC+1:LOCATE 0,10,0:COLOR2:PRINT SC
215 ON STICK(0)GOTO 220,230,230,230,220,250,250,25
0
220 GOTO 300
230 IF R<32 THEN PSET(R-1,23)CHR$(127),0:R=R+1:PSE
T(R+1,23)CHR$(127),4:GOTO 300
240 GOTO 300
250 IF R>8 THEN PSET(R+1,23)CHR$(127),0:R=R-1:PSET
(R-1,23)CHR$(127),4
300 V=BH:W=BV:BH=V+AH:BV=W+AV
310 ON POINT(BH*8,BV*8) GOTO ,,,360,,,370
320 PSET(V,W)CHR$(127),0:PSET(BH,BV)GR$(0),4:IF BV
>22 THEN GOTO 500 ELSE IF BH<>14 OR BH<>5 THEN GOT
O 210
```

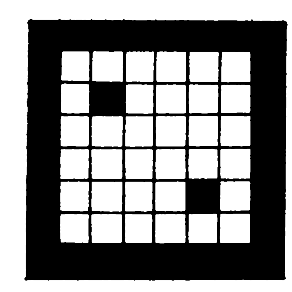
```
330 IF AV<0 AND BH>9 AND BH<31 THEN V=BH:BH=BH+INT
(RND*2):PSET(V,BV)CHR$(127),0:PSET(BH,BV)GR$(0),4
340 GOTO 210
360 BH=V:BV=W:AV=-SGN(AV):GOTO 210
370 BH=V:BV=W:AH=-SGN(AH):GOTO 210
500 COLOR 2
510 FOR I=1 TO 20
520 LOCATE 0,24:COLOR2:PRINT"APPUYER SUR ACTION PO
UR REJOUER";CHR$(11)
530 IF STRIG(0) THEN GOTO 30
540 IF INKEY$="S" THEN CLS:END
550 NEXT I
560 FOR I=1 TO 20
570 LOCATE 0,24:COLOR2:PRINT"APPUYER SUR   S    PO
UR STOPPER";CHR$(11)
580 IF STRIG(0) THEN GOTO 30
590 IF INKEY$="S" THEN CLS:END
600 NEXT I
610 GOTO 510
1000 'TERRAIN
1010 SCREEN ,0,0:CLS
1020 BOXF(6,0)-(34,24)CHR$(127),0
1030 LINE(6,0)-(34,0)CHR$(127),4
1040 LINE(6,1)-(6,23)CHR$(127),7
1050 LINE(34,1)-(34,23)CHR$(127),7
1060 LINE(10,10)-(15,10)CHR$(127),4
1070 LINE(30,10)-(25,10)CHR$(127),4
1130 LINE(19,23)-(21,23)CHR$(127),4
1140 BH=8+INT(RND*24):BV=9+INT(RND*10)
1150 PSET(BH,BV)GR$(0),4
1160 LOCATE 0,9,0:COLOR 2:PRINT "SCORE"
1170 LOCATE 0,15,0:PRINT "HIGHT"
1180 IF MS<SC THEN MS=SC
1190 LOCATE 0,16,0:PRINT MS:SC=0:LOCATE 0,10,0:PRI
NT SC
1200 RETURN
3000 'PRESENTATION
3010 GOSUB 1000:BOXF(0,0)-(5,24)CHR$(127),0
3120 ATTRB 1,1:COLOR 1
3140 LOCATE 15,11,0:PRINT"SQUASH"
3160 ATTRB 0,0:COLOR 2
3170 LOCATE 0,24:PRINT"APPUYER SUR ENTREE POUR COM
MENCER";CHR$(11)
3180 IF INKEY$<>CHR$(13)THEN GOTO 3180
3190 RETURN
```

Annexes

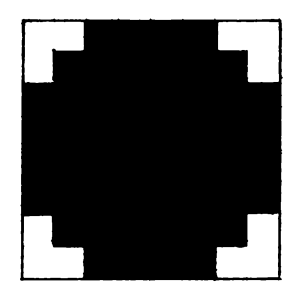
A. — RECUEIL DE CARACTÈRES DÉFINIS

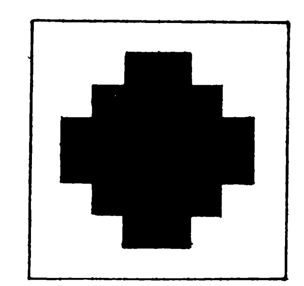
* Dés à jouer



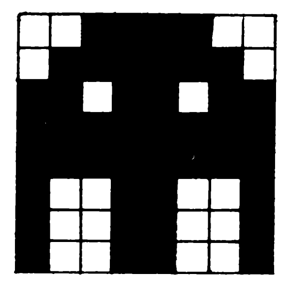


* Pions

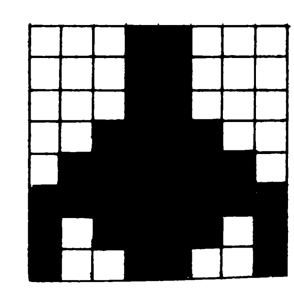




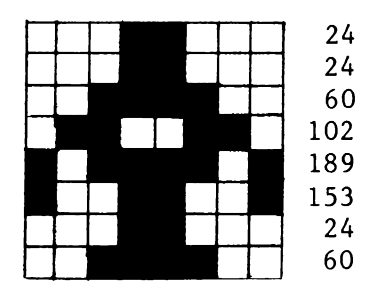
* Envahisseur



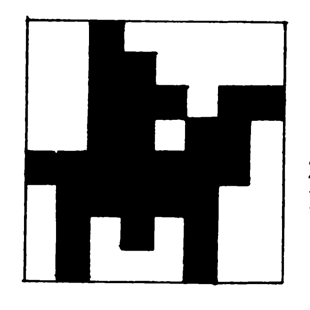
* Fusée

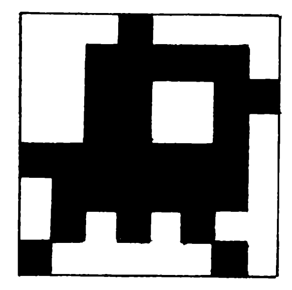


* Missile

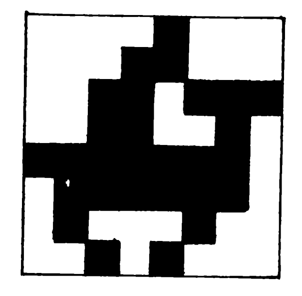


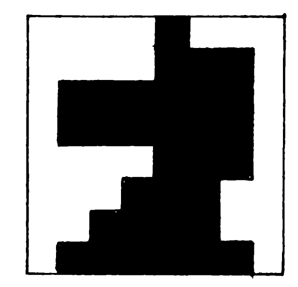
* Chevaux (TIERCE)



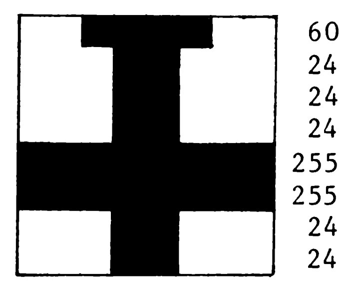


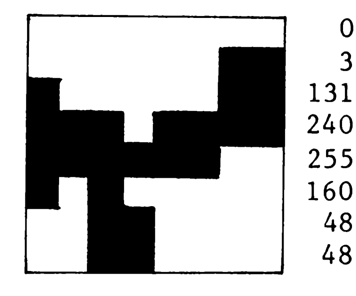
* Cavalier



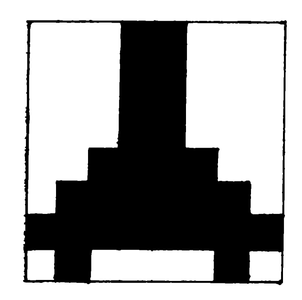


* Avions (DCA)

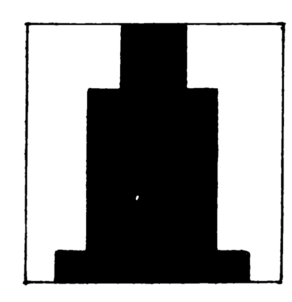




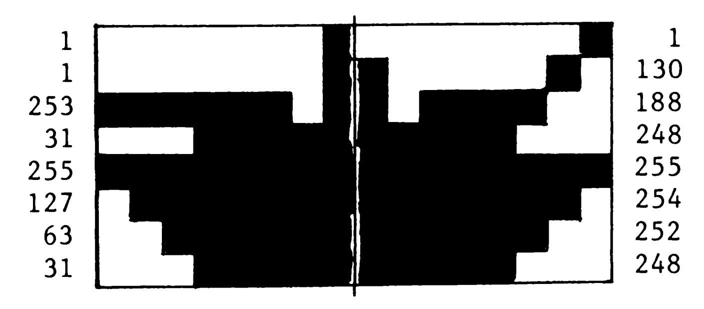
* Batterie de DCA



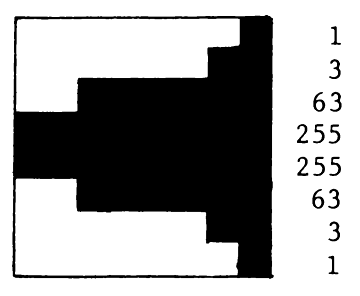
* Obus

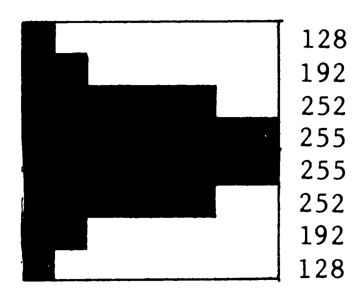


* Bateau (BOMBARDIER)

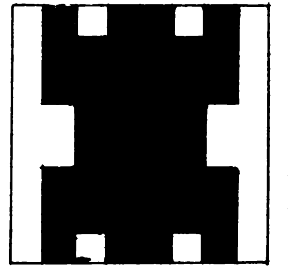


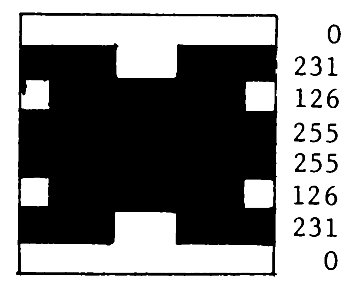
* Bombes





*Autos (STOCK CAR)





B. — CODES ASCII

ASCII Code	Character	ASCII Code	Character	ASCII Code	Character
000	NUL	036	$	072	H
001	SOH	037	%	073	I
002	STX	038	&	074	J
003	ETX	039	'	075	K
004	EOT	040	(	076	L
005	ENQ	041	)	077	M
006	ACK	042	*	078	N
007	BEL	043	+	079	O
008	BS	044	,	080	P
009	HT	045	-	081	Q
010	LF	046	.	082	R
011	VT	047	/	083	S
012	FF	048	0	084	T
013	CR	049	1	085	U
014	SO	050	2	086	V
015	SI	051	3	087	W
016	DLE	052	4	088	X
017	DC1	053	5	089	Y
018	DC2	054	6	090	Z
019	DC3	055	7	091	[
020	DC4	056	8	092	\
021	NAK	057	9	093	]
022	SYN	058	:	094	↑
023	ETB	059	;	095	<
024	CAN	060	<	096	'
025	EM	061	=	097	a
026	SUB	062	>	098	b
027	ESCAPE	063	?	099	c
028	FS	064	@	100	d
029	GS	065	A	101	e
030	RS	066	B	102	f
031	US	067	C	103	g
032	SPACE	068	D	104	h
033	!	069	E	105	i
034	"	070	F	106	j
035	#	071	G	107	k

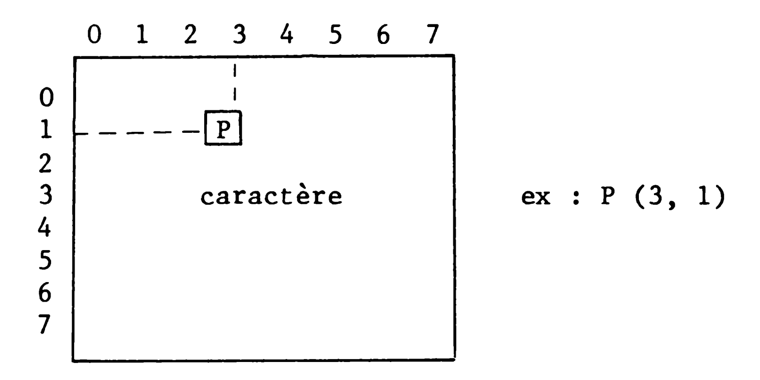
<table>
<tr><td>108</td><td>l</td><td>115</td><td>s</td><td>122</td><td>z</td></tr>
<tr><td>109</td><td>m</td><td>116</td><td>t</td><td>123</td><td>|</td></tr>
<tr><td>110</td><td>n</td><td>117</td><td>u</td><td>124</td><td>|</td></tr>
<tr><td>111</td><td>o</td><td>118</td><td>v</td><td>125</td><td>|</td></tr>
<tr><td>112</td><td>p</td><td>119</td><td>w</td><td>126</td><td>~</td></tr>
<tr><td>113</td><td>q</td><td>120</td><td>x</td><td>127</td><td>DEL</td></tr>
<tr><td>114</td><td>r</td><td>121</td><td>y</td><td></td><td></td></tr>
</table>

ASCII codes are in decimal.
LF=Line Feed, FF=Form Feed, CR=Carriage Return, DEL=Rubout

C. — INSTRUCTIONS GRAPHIQUES ET SONORES

I. Correspondance entre coordonnées en mode graphique et coordonnées en mode caractére

Soit le point P de coordonnées (H, V) à l'intérieur d'un caractère ($0 \leq H \leq 7$, $0 \leq V \leq 7$) :



Si les coordonnées (en mode caractère) du caractère sont (A, B) ; les coordonnées (X, Y) en mode graphique du point P sont :

(X, Y) = (A * 8 + H, B * 8 + V)

II. Instructions graphiques

* ATTRB A, B, C

Doublement des dimensions des caractères et masquage (démasquage voir UNMASK).

A = 1 : largeur doublée   A = 0 : largeur normale

B = 1 : hauteur doublée   B = 0 : hauteur normale

C = 1 : caractères noirs sur fond noir (mode masqué)

C = 0 : mode normal

* BOX $(X_1, Y_1) - (X_2, Y_2)$, A

Rectangle (périmètre) en mode graphique.

* BOXF $(X_1, Y_1) - (X_2, Y_2)$, A

Rectangle plein en mode graphique.

$0 \leqslant X_i \leqslant 319$ et $0 \leqslant Y_i \leqslant 199$

A : couleur des points (voir annexe D.1)

$0 \leqslant A \leqslant 7$ pour une couleur de graphisme

$-8 \leqslant A \leqslant -1$ pour une couleur de fond

graphisme		fond	
0	noir	- 1	noir
1	rouge	- 2	rouge
2	vert	- 3	vert
3	jaune	- 4	jaune
4	bleu	- 5	bleu
5	magenta	- 6	magenta
6	cyan	- 7	cyan
7	blanc	- 8	blanc

* BOX $(X_1, Y_1) - (X_2, Y_2)$ "caractère", A, B

Rectangle (périmètre) en mode caractère.

* BOXF $(X_1, Y_1) - (X_2, Y_2)$ "caractère", A, B

Rectangle plein en mode caractère.

(ex : "C", C$, CHR$, GR$)

A est la couleur de forme (voir annexe D.2)

B est la couleur de fond (voir annexe D.2)

* CLS

Efface la fenêtre de travail.

* COLOR A, B

Définit la couleur des prochains caractères.

A est la couleur de forme (voir annexe D.2)

B est la couleur de fond (voir annexe D.2)

* CONSOLE A, B

Définit la hauteur de la fenêtre de travail.

A : N° de ligne supérieure $0 \leqslant A \leqslant 24$)
) A B
B : N° de ligne inférieure $0 \leqslant B \leqslant 24$)

* CSRLIN

Donne le numéro de ligne du curseur

LINE (X_1, Y_1) - (X_2, Y_2) A :

Trace une ligne en mode graphique entre (X_1, Y_1) et (X_2, Y_2) de couleur A (voir annexe D.1).

* LINE $(X_1, Y_1) - (X_2, Y_2)$ "caractère", A, B

Trace une ligne en mode caractère (voir annexe D.2 et BOX).

* LOCATE X, Y, Z

Positionne le curseur sur l'écran en (X, Y) (mode caractère).

X est la coordonnée horizontale $0 \leq X \leq 39$

Y est la coordonnée verticale $0 \leq Y \leq 24$

Z détermine la présence ou non du curseur

Z = 0 : curseur invisible   Z = 1 : curseur visible

* POINT (X, Y)

Donne la couleur du point de coordonnées (X, Y) en mode graphique (0   X   319 et 0   Y   199). Le code des couleurs est donné dans l'annexe D.1.

* POS

Donne le numéro de ligne où se trouve le curseur.

* PSET (X, Y), A

Positionne un point de coordonnées graphiques (X, Y) et de couleur A (0   X   319 et 0   Y   199). A est donné dans l'annexe D.1.

* PSET (X, Y), "caractère", A, B

Positionne un caractère dans la case de coordonnées (X, Y) en mode caractère. "caractère", A et B ont la même signification que pour l'instruction BOX en mode caractère (voir annexe D.2).

* SCREEN (X, Y)

Donne le code ASCII du caractère de coordonnées (X, Y) (en mode caractère). ($0 \leq X \leq 39$ et $0 \leq Y \leq 24$).

* SCREEN A, B, C

Modifie les couleurs de la fenêtre de travail et du cadre.

A couleur de forme)
B couleur de fond) voir annexe D.2
C couleur du cadre)

* UNMASK

Enlève dans la fenêtre de travail le masque des caractères cachés par ATTRB.

III. Instruction sonore

* PLAY expression chaîne

Joue l'air codé par expression chaîne .

Les notes sont DO, RE, MI, FA, SO, LA, SI, avec éventuellement dièze (♯) ou bémol (♭) et les arguments :

Attaque :

Noté A, doit être suivi d'une valeur comprise entre 0 et 255 (amortissement de la note).

Longueur :

Noté L, doit être suivi d'une valeur comprise entre 1 et 96 (durée de la note).

Octave :

Noté O, doit être suivi d'une valeur comprise entre 1 et 5 (5 est l'octave la plus aiguë).

Tempo :

Noté T, doit être suivi d'une valeur comprise entre 1 et 255 (vitesse d'exécution).

Le silence est noté P.

D. — CODES COULEURS

I. Mode graphique

couleurs formes		couleurs fond	
0	noir	- 1	noir
1	rouge	- 2	rouge
2	vert	- 3	vert
3	jaune	- 4	jaune
4	bleu	- 5	bleu
5	magenta (violet)	- 6	magenta
6	cyan (bleu clair)	- 7	cyan
7	blanc	- 8	blanc

II. Mode caractère

0 noir
1 rouge
2 vert
3 jaune
4 bleu
5 magenta
6 cyan
7 blanc

LA BIBLIOTHEQUE EDIMICRO

Collection " Guides microordinateurs "

de MERLY	GUIDE DE L'APPLE	Tome 1 L'APPLE standard Tome 2 Les extensions Tome 3 Les applications
BAYVEJIEL	GUIDE DE L'ORIC	
BIEBER, PERBOST RENUCCI	GUIDE DU TO 7	

Collection " Jeux "

CHANE-HUNE, DARBOIS	JEUX SUR ORIC
PERBOST, RENUCCI	JEUX SUR TO 7

Collection " Logiciels "

BONNET, DINH	MULTIPLAN SUR APPLE Exercices de Gestion

EDIMICRO
121-127, avenue d'Italie, 75013 PARIS

Imprimé en France. — JOUVE, 18, rue Saint-Denis, 75001 PARIS
N° 11705. Dépôt légal : Septembre 1983